Stochastic Lagrangian Relaxation in Power Scheduling of a Hydro-Thermal System under Uncertainty

D I S S E R T A T I O N

zur Erlangung des akademischen Grades
doctor rerum naturalium
(Dr. rer. nat.)
im Fach Mathematik

eingereicht an der
Mathematisch-Naturwissenschaftlichen Fakultät II
Humboldt-Universität zu Berlin

von
Herrn Dipl.-Math. Matthias Peter Nowak
geboren am 7. Mai 1967 in Berlin-Mitte

Präsident der Humboldt-Universität zu Berlin:
Prof. Dr. Dr. h.c. Hans Meyer

Dekan der Mathematisch-Naturwissenschaftlichen Fakultät II:
Prof. Dr. Bodo Krause

Gutachter:

1. Prof. Dr. John R. Birge
2. Prof. Dr. Werner Römisch
3. Prof. Dr. Rüdiger Schultz

eingereicht am:	23. Dezember 1999
Tag der mündlichen Prüfung:	1. Dezember 2000

Abstract

We consider a power generation system comprising thermal units and pumped hydro storage plants, and introduce a model for its weekly cost-optimal operation. Due to the uncertainty of the load, the mathematical model represents a dynamic (multi-stage) stochastic program. The model involves a large number of mixed-integer (stochastic) decisions but its constraints are loosely coupled across operating power units. The coupling structure is used to design a stochastic Lagrangian relaxation method, which leads to a decomposition into stochastic single unit subproblems.

The stochastic subproblems have deterministic counterparts, which makes it easy to develop algorithms for the stochastic problems. In this paper, a descent method for stochastic storage problems and an extension of dynamic programming towards stochastic programs are developed.

The solution of the dual problem provides multipliers leading to preferred schedules (binary primal variables). The crossover heuristics evaluates the economic dispatch problems corresponding to a sequence of such preferred schedules. The combination of the restriction on dual preferred schedules (Lagrangian reduction) with the evaluation of a sequence (facet search) leads to an efficient method. The numerical results on realistic data of a German utility justify this approach.

Keywords:
Stochastic multi-stage mixed-integer programming, Stochastic Lagrangian relaxation, Duality of mixed-integer programs, Unit commitment

1991 MSC: 49M10 49M27 49N15 90C06 90C11 90C15 90C27 90C35 90C39

Zusammenfassung

Wir betrachten ein Kraftwerkssystem mit thermischen Blöcken und Pumpspeicherwerken und entwickeln dafür ein Modell für den kostenoptimalen Wochenbetrieb. Auf Grund der Ungewißheit des Bedarfs an elektrischer Energie ist das mathematische Modell ein mehrstufiges stochastisches Problem. Dieses Modell beinhaltet viele gemischt-ganzzahlige stochastische Entscheidungsvariablen. Die Variablen einzelner Einheiten sind aber nur durch wenige Nebenbedingungen miteinander verbunden, welches die Zerlegung in stochastische Teilprobleme erleichtert.

Diese stochastischen Teilprobleme besitzen deterministische Analoga, deren Lösungsverfahren entsprechend erweitert werden können. In dieser Arbeit werden ein Abstiegsverfahren für stochastische Speicherprobleme und eine Erweiterung der dynamischen Programmierung auf stochastische Probleme betrachtet.

Die Lösung des dualen Problems führt zu Schattenpreisen, die bestimmte Einsatzentscheidungen bevorteilen. Die Heuristik zur Suche von primalen zulässigen Punkten wertet eine Folge von zugeordneten Economic-Dispatch-Problemen aus. Die Kombination der Einschränkung auf dual bevorzugte Fahrweisen (Lagrangian reduction) mit der Auswertung einer Folge von Economic-Dispatch-Problemen (Facettensuche) führt zu einem effizienten Verfahren. Die numerischen Ergebnisse an Hand realistischer Daten eines deutschen Versorgungsunternehmens rechtfertigen diesen Zugang.

Schlagwörter:
Stochastische mehrstufige gemischt-ganzzahlige Optimierung, Stochastische Lagrange-Relaxation, Dualität gemischt-ganzzahliger Optimierungsprobleme, Optimierung von Kraftwerkssystemen

1991 MSC: 49M10 49M27 49N15 90C06 90C11 90C15 90C27 90C35 90C39

Preface

This doctoral thesis is an outgrowth of my work together with my colleagues at the Department of Mathematics at Humboldt University of Berlin. It treats some aspects of short-term optimization in power scheduling with thermal units, pumped hydro storage plants, and uncertain demand. The resulting optimization problem presents a mixed-integer multi-stage stochastic linear program. The approach "stochastic Lagrangian relaxation" decomposes this program into a number of subproblems and a non-differentiable concave dual problem. The developed algorithms for solving these special structured subproblems and a Lagrange heuristics constitute the main content of this thesis.

First of all I would like to thank Prof. Werner Römisch for his support through all periods of this work. Although this work was part of a larger research project under his leadership, I got the liberty to pursue some ideas for myself. He also arranged contacts with other research groups in stochastic programming, which contributed to the work. And I have to thank him for providing me excellent working conditions and computing facilities.

The work presented in this thesis was made possible mainly through a grant from the Graduiertenkolleg "Geometrie und nichtlineare Analysis" at Humboldt University, which is gratefully acknowledged.

This thesis was completed while I was supported by a joint grant of Professors Werner Römisch and Rüdiger Schultz (Gerhard-Mercator-Universität Duisburg) "Unit commitment in power generation planning" (German Federal Ministry of Education, Science, Research, and Technology). I wish to extend my gratitude to Professor Rüdiger Schultz who helped me in various matters several times. This thesis has also benefitted from the grant "Online power scheduling under uncertainty" by the German Research Foundation awarded to Professors Römisch and Schultz.

During this Ph.D. work I had the pleasure of staying 3 months at the "International Institute for Applied Systems Analysis" in Laxenburg (Austria). Within the "Young Scientists Summer Program 1996" I worked in the group "Optimization under Uncertainty" under supervision of Prof. A. Ruszczyński and Prof. G.Ch. Pflug, to whom I am indebted for stimulating discussions. I had also the pleasure of discussing related topics with scientists from all over the world. And

I made friends for my life[1].

The numerical results presented in Chapter 4 were not possible without the FORTRAN-code NOA 3.0 of Prof. Kiwiel, because the applicability of a decomposition approach mainly depends on the availability of an efficient solver for the non-differentiable master problem.

Many colleagues have been generous in giving me the benefit of their mathematical and computational experience. Foremost I wish to thank Andris Möller for many hints about the combination of Fortran and C, how to avoid instabilities and about numerical properties of certain algorithms. The work together with C.C. Carøe, N. Gröwe-Kuska, and I. Wegner provided me with insights in different parts of stochastic programming. The discussions with S. Jaschke contributed to a great extend to my knowledge about stochastic programs. And a valuable comment by Prof. J. Dupačová made some proofs simpler. With S.-E. Fleten and Dr. A. Gjelsvik from Norway I discussed the descent method for stochastic programs which resulted in substantial extensions of that method. I am grateful to my colleagues of the Department of Mathematics at Humboldt University Berlin who have shown interest in my work and have made the working atmosphere so stimulating.

Finally, I would like to honour the work of the GNU community for the great software I used. The combinations of Linux, g++, vim, make, gnuplot, and LaTeX formed an environment, which made my work pretty comfortable.

And last but not least, I thank all my friends for being there when life is so painful.

[1] IIASA is also known for creating relationships – E. & J.: I wish you all the best!

Overview

This thesis focuses on the numerical solution of the power scheduling in a hydro-thermal system with the additional aspect of uncertain demand. Due to the presence of pumped hydro storage plants the time horizon comprises about one week. Since the optimization should provide hourly scheduling decisions, an hourly time discretization was taken. The uncertainty of the demand is modeled as a stochastic process. This modeling led to a mixed-integer multi-stage stochastic linear program.

Chapter 1 provides the notion of a multi-stage stochastic linear program based on stochastic processes and filtrations. In case of a finite number of time periods and scenarios, such processes correspond to scenario trees. A definition of a scenario tree is developed, which combines graph-theoretical and stochastic aspects. This notion of a scenario tree forms the basement for the descriptions of algorithms and proofs in all other chapters. After a short overview over decomposition schemes in section 1.6, two primal decomposition methods, Nested Benders Decomposition (in section 1.7) and Stochastic Dual Dynamic Programming (in section 1.8) are presented.

The solution method applied to the power scheduling problem is developed in Chapter 2. Section 2.2 concerns the stochastic Lagrangian relaxation method in detail. A major advantage of this method is the common measurability of the decision process and the process of the stochastic multipliers. The presence of integer variables leads to a duality gap (in section 2.4). Sections 2.5 and 2.6 discuss how the Lagrange parameters can be used in order to obtain primal points with a reasonably good objective value.

Chapter 3 concentrates on the development of efficient methods for the stochastic subproblems. Section 3.1 contains the description of a descent method for stochastic storage problems while section 3.2 shows the modifications of the dynamic programming algorithm that are necessary in order to solve stochastic problems.

The numerical methods are tested on realistic data from a German utility, and the tests are reported in Chapter 4. Section 4.5 compares, on deterministic problems, the heuristics developed in this thesis with the Zhuang/Galiana heuristics that was modified in order to solve hydro-thermal problems.

Contents

Preface **4**

Overview **6**

1 Introduction **9**
1.1 Optimization under Uncertainty 10
1.2 Stochastic Processes 12
1.3 Nonanticipativity 13
1.4 Scenario Tree 15
1.5 Multi-Stage Stochastic Linear Programs 17
1.5.1 Deterministic LP-equivalent 18
1.5.2 Deterministic Graph-Equivalent 20
1.6 Decomposition Schemes 21
1.7 Nested Benders Decomposition 22
1.8 Stochastic Dual Dynamic Programming 27

2 Solving the Power Scheduling Problem **31**
2.1 Model of a Power Generation System 31
2.2 Stochastic Lagrangian Relaxation 36
2.2.1 Lagrangian Relaxation 37
2.2.2 Stochastic Lagrangian Relaxation for Dynamic Recourse Problems 38
2.3 Decomposition 41
2.4 Duality Gap in Mixed-Integer Programs 43
2.5 Lagrangian Reduction 46
2.6 Facet Search 50

3 Algorithms for Stochastic Subproblems **55**
3.1 Descent Method for Stochastic Storage Problems 55
3.1.1 The Model 55
3.1.2 Example for a Simple Direction 57
3.1.3 Conditions on Directions to be Descent Directions 59
3.1.4 Extensions 64

3.1.5 Algorithm 68
3.2 Dynamic Programming for Stochastic Programs 69

4 Application to a Hydro-Thermal Power Generation System 74
4.1 Kiwiel's Proximal Bundle Method 74
4.2 Descent Method for the Storage Subproblems 77
4.3 Stochastic Dynamic Programming 78
4.4 Lagrangian Reduction and Facet Search 85
4.5 Comparison with a modified Zhuang/Galiana Heuristics 91
4.6 Economic Dispatch 94
4.7 Numerical Results 99

Conclusions 107

Notations 109

Chapter 1

Introduction

For many years energy optimization has dealt with large scale mixed-integer linear programs. This thesis considers a program that can be used for controlling a real generation system comprising thermal power units and pumped hydro storage plants. Moreover, it should be possible to use the solution of that program in order to control the system in a least cost manner. Therefore, the program should be solved in real time.

The energy optimization problem becomes more complicated, if the uncertainty of the demand is taken into account. On the other hand, this assumption makes the problem more realistic in case of short term planning. Uncertainty means, that data about future realizations are not exactly known. A forecast method can provide data within a certain accuracy, while historic data provides information about the distribution of the forecast error.

The stochastic nature introduces a new dimension — the mathematical model describes the process that has to be optimized, but also the time periods at which the decision maker can make decisions, and it should take the rolling time horizon into account. In a deterministic framework, the input data were taken as they are. The output of the optimization aided the decision maker. The question of the dependency of the output on the accuracy of the input led to parametric optimization and perturbation analysis. The hope was that the output could approximate necessary decisions even for slightly perturbed input data.

However, in a stochastic framework, more information about input data is involved. In models including a forecast, the information about random data decreases with increasing distance to the current time period. The progress in time determines the additional amount of information, which is available to the decision maker. Each decision should be made as late as possible and with the information obtained up to this time period. Hence, the "real world" background determines:

- the correspondence of decisions to time periods according to their influence on state variables,

- their assignment to time periods, they have to be made at,
- the availability of information,
- the degree, which the output of the optimization is used to.

These points need a discussion in order to classify a model and its application as an operational or as a planning model, which is beyond the scope of this thesis. However, random variables appear in the underlying optimization problem of such models. These random variables are either stochastic input data or decision variables. The latter are flexible up to a certain degree with respect to the random input data. The optimization of programs including random variables is the challenge of stochastic programming.

1.1 Optimization under Uncertainty

Optimization can be realized as picking the best choice among several others. Sometimes, the values of all possibilities are not known for a certainty. Even in such situations one would like to try "the best". At this point, people react differently according to their understanding of loss and revenue.

If the average outcome should be maximized, then the mathematical model reads:

$$\max_{x \in X} \mathbb{E} g_0(x, \omega) \tag{1.1}$$

subject to:

$$\forall \omega \in \Omega, i = 1, \ldots, m \; : \; g_i(x, \omega) \leq 0 \, . \tag{1.2}$$

In this model Ω denotes the abstract space of random events. These random events do not depend on the decision x. The expectation $\mathbb{E} g_o(x, \omega)$ bases on the measure μ, which is a probability measure with respect to the probability space $(\Omega, \mathcal{A}, \mathbb{P})$. An element ω of this space is called *scenario*. A *realization* of a random variable $\boldsymbol{d}$[1] is denoted by $d(\omega)$.

The solution is called *feasible* with respect to scenario ω, if it meets all constraints, i.e: $g_i(x, \omega) \leq 0$ for all $i = 1, \ldots, m$. If the solution meets the constraints for all scenarios in Ω, then the solution is admissible.

Such a constraint can be weakened in the sense, that the decision is given the permission to fail in unlikely situations. If a solution is requested, which is admissible for a certain percentage of all situations, then we are in the field of *chance constrained stochastic programming*.

The formulation given above is to general in order to be suitable for an application. In the linear case the functions $g_i, \; i = 0, \ldots, m$ are replaced by:

$$g_0(x, \omega) := c^T(\omega) x \tag{1.3}$$

$$g_i(x, \omega) := (A_{i,\cdot}) x - b_i(\omega) \, , \; i = 1, \ldots, m, \tag{1.4}$$

[1] Bold symbols denote the random variables without mentioning the event explicitly.

where $(A(\omega), b(\omega), c(\omega))$ is a realization of the random event ω. The tuple $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ presents a random variable as defined next.

Uncertain data as well as information dependent decisions are modeled as random variables. A real-valued random variable x can be considered as a function

$$x : (\Omega, \mathcal{F}, \mathbb{P}) \to (\mathbb{R}, \mathcal{B}), \tag{1.5}$$

where $(\Omega, \mathcal{F}, \mathbb{P})$ is a suitable probability space. Using random variables the linear model reads:

$$\max_{x \in X} \mathbb{E}\, \boldsymbol{c}^T x \tag{1.6}$$

subject to:

$$\boldsymbol{A}x \leq \boldsymbol{b}\,. \tag{1.7}$$

Sometimes, one has to make a decision with known costs before the observation of the unknown is made, and one gets the opportunity to correct the decision after the event realizes, but this time the costs are scenario dependent. The cost of this correction is called recourse cost. Such a recourse model in a linear case reads:

$$\min_x c^T x + \mathbb{E}Q(x, \omega) \tag{1.8}$$

subject to:

$$\begin{aligned} Ax &\leq b && (1.9)\\ Q(x,\omega) &:= \min_{y(\omega) \in Y} d(\omega)^T y(\omega) && (1.10)\\ T(\omega)x + W(\omega)y(\omega) &\leq h(\omega)\,, && (1.11) \end{aligned}$$

or short as:

$$\min \left\{ c^T x + \mathbb{E} \min \left\{ \boldsymbol{d}^T \boldsymbol{y} \,|\, \boldsymbol{y} \in Y,\ \boldsymbol{T}x + \boldsymbol{W}\boldsymbol{y} \leq \boldsymbol{h} \right\} \,|\, x \in X,\ Ax \leq b \right\}\,. \tag{1.12}$$

The decision variables x and $\boldsymbol{y}$ are called first-stage and second-stage variables, respectively. The model itself is also known as a two-stage model.

If the first-stage decision represents a risky investment, then the recourse costs would reach a value in certain scenarios, which could lead to a financial disaster for the investor. Therefore, the minimization of the average costs is not suitable for this underlying real-world problem. Hence, the investor is interested in minimizing the average cost and the variation of the recourse costs, too. This leads to optimization models involving the second moment of the second-stage variables. However, this thesis concentrates on problems with moderate recourse costs.

The two-stage model presented above describes real-world problems, where decisions are distinguished by the fact, whether they are made with or without the knowledge of the unknown. Often, the information about future realizations is not available all at once — moreover, the observations correspond to time

periods. Then, a finer granularity of the time periods describes the problem which yields a multi-stage stochastic problem. The part of random variables is taken by stochastic processes.

1.2 Stochastic Processes

In two-stage stochastic programs variables are distinguished by the fact, whether they are stochastic or not. Such a simple distinction is not possible, if the values of the random variables are revealed at different time periods. On the one hand, stochastic parameters correspond to time periods, when their values are known. This behavior is modeled by stochastic processes. The decision maker is faced with the difficulty to make decisions without prior knowledge of future realizations of these random variables. Usually, the decisions are related to time periods, when they have to be made and with the knowledge obtained up to and including this time period. This leads to a time structure of the problem, and the decision process is a stochastic process.

Usually, stochastic processes as the Gaussian process comprise a continuous time structure. However this paper only deals with stochastic programming models that allow decisions at the end of certain time periods. The stochastic input data are also taken at certain time period. Therefore, all stochastic processes are considered as discrete time stochastic processes in this thesis.

Definition 1.1 (Discrete time stochastic process) A stochastic process x is a random variable with a time structure, i.e.:

$$x : \Omega \times \{1, \ldots, T\} \longrightarrow \mathbb{R}\,. \tag{1.13}$$

The components $x(.,t)$ act as osdinary random variables with the measurability:

$$x(.,t) : (\Omega, \mathcal{A}_t, \mathbb{P}) \to (\mathbb{R}, \mathcal{B})\,. \tag{1.14}$$

For the aim of simplicity, the bold notation $\boldsymbol{x}_t$ is used, if the first argument ω is omitted (i.e. $\boldsymbol{x}_t := x(.,t)$), while $\boldsymbol{x}$ denotes the vector of values $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$.

Usually, stochastic processes show some structure. Looking at the values of $\boldsymbol{x}_t$ some random events ω and ν can not be distinguished up to a certain time period t_0:

$$\forall t = 1, \ldots, t_0 \;:\; x(\omega, t) = x(\nu, t)\,. \tag{1.15}$$

Hence, the random parameters of the stochastic program provide an information structure. This information structure is usually represented by an increasing sequence of σ-fields, which are generated by random parameters. They form a filtration as defined next.

Definition 1.2 (Filtration) Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space. The family of σ-fields $\mathcal{F} = \{\mathcal{A}_t\}_{t=1}^T$ is called a filtration, if:

- $\mathcal{A}_t \subseteq \mathcal{A}$ and $\mathcal{A}_t$ is a σ-field,
- $\forall t = 2, \ldots, T \; : \; \mathcal{A}_{t-1} \subseteq \mathcal{A}_t$.

Each process $\boldsymbol{x}$ is measurable with respect to the filtration $\tilde{\mathcal{F}} = \{\tilde{\mathcal{A}}_t\}_{t=1}^T$ with $\tilde{\mathcal{A}}_t = \mathcal{A}$, because each $\boldsymbol{x}_t$ was measurable with respect to $\mathcal{A}$. But this filtration is too big in order to contain any information provided by the stochastic process. The interesting question rises, does there exists a smaller filtration such, that the stochastic process is measurable to it. The adapted filtration should contain small σ-fields at the beginning ($t = 1$), and for subsequent time periods these σ-fields should grow as much as new information gets available. Such a filtration is generated using the stochastic process as a base for creating the σ-fields.

Definition 1.3 A stochastic process $\boldsymbol{x}$ as defined in 1.1 determines the corresponding filtration by:

$$\mathcal{F} := \{\mathcal{A}_t\}_{t=1}^T \tag{1.16}$$

$$\mathcal{A}_t := \sigma\{A \subseteq \Omega | \exists \{B_\tau\}_{\tau=1}^t \subseteq \mathcal{B} \; : \; A = \cap_{\tau=1}^t (x_\tau)^{-1}(B_\tau)\} \, . \tag{1.17}$$

The filtration $\mathcal{F}$ is generated by the process $\boldsymbol{x}$, which is denoted by $\mathcal{F} = \mathcal{F}(\boldsymbol{x})$.

This filtration contains the information, that can be obtained by observing $\boldsymbol{x}$. At the beginning nothing is known about the random event. With each time period the set of possible scenarios is reduced to the set $P_t = \cap\{A \in \mathcal{A}_t | \omega \in A\}$, where ω is the realizing random scenario. It is no longer likely that other scenarios get real, since they should have shown a different behavior during the past.

1.3 Nonanticipativity

In a mathematical framework, stochastic programs are assumed to contain all available information as input data. Some of these data are stochastic processes. The decision maker is faced with the problem to find the best decision with the knowledge about the stochastic process obtained so far. Each observation encloses the set of possible future realizations. Generally, this set is not a singleton. Therefore, the requested decision should minimize[2] the expected objective value with respect to this set.

Because the decision maker can not look ahead, this information constraint is called "nonanticipativity constraint". A rigorous discussion of nonanticipativity and related topics can be found in [RW76]. Since the information of a stochastic

[2] Or maximize the revenue.

process is contained in the corresponding filtration, filtrations can be used to define the nonanticipativity of a decision process.

Definition 1.4 (Nonanticipativity) A stochastic process $\boldsymbol{x}$ is nonanticipative with respect to the filtration $\mathcal{F} = \{\mathcal{A}_t\}_{t=1}^{T}$, if $\boldsymbol{x}_t$ is measurable with respect to $\mathcal{A}_t$ for all $t = 1, \ldots, T$.

The restriction on a process to be nonanticipative with respect to the stochastic input data can be expressed using the filtration, which is generated by the process itself.

Proposition 1.1 Let $\mathcal{F}(\boldsymbol{d})$ denote the filtration generated by the stochastic process $\boldsymbol{d}$, while $\boldsymbol{x}$ denotes the decision process of a stochastic program. This stochastic process generates the filtration $\mathcal{F}(\boldsymbol{x})$. The inclusion with respect to filtrations is defined as:

$$\mathcal{F}^1 \subseteq \mathcal{F}^2 \Longleftrightarrow \forall t = 1, \ldots, T \,:\, \mathcal{A}_t^1 \subseteq \mathcal{A}_t^2 \tag{1.18}$$

$\Longrightarrow$ The process $\boldsymbol{x}$ is nonanticipative with respect to $\mathcal{F}(\boldsymbol{d})$, if and only if the following inclusion holds:

$$\mathcal{F}(\boldsymbol{x}) \subseteq \mathcal{F}(\boldsymbol{d})\,. \tag{1.19}$$

Proof: Assume for the first part, that $\boldsymbol{x}$ is nonanticipative with respect to the filtration generated by $\boldsymbol{d}$, i.e. $\boldsymbol{x}$ is measurable with respect to $\mathcal{F}(\boldsymbol{d})$. For all $B \in \mathcal{B}(\mathbb{R}^n)$, $(\boldsymbol{x}_t)^{-1}(B) \in \mathcal{A}_t(\boldsymbol{d})$ is fulfilled. Assuming $\mathcal{A}_{t-1}(\boldsymbol{x}) \subseteq \mathcal{A}_{t-1}(\boldsymbol{d})$, the inclusion $\mathcal{A}_t(\boldsymbol{x}) \subseteq \mathcal{A}_t(\boldsymbol{d})$ holds, because $(\boldsymbol{x}_t)^{-1}(B)$ was among the generating elements of $\mathcal{A}_t(\boldsymbol{x})$. The recursion over t yields the inclusion for all time periods. Therefore $\mathcal{F}(\boldsymbol{d})$ comprises $\mathcal{F}(\boldsymbol{x})$. Hereby, $\mathcal{B}(\mathbb{R}^n)$ denotes the Borel-sets of $\mathbb{R}^n$, and $\mathcal{A}_t(y)$ is used as an abbreviation for the t-component of $\mathcal{F}(y)$.

Next, the measurability of $\boldsymbol{x}$ with respect to $\mathcal{F}(\boldsymbol{d})$ is shown. The origin of any $B \in \mathcal{B}(\mathbb{R}^n)$ with respect to $\boldsymbol{x}_t$ is a generating element of $\mathcal{A}_t(\boldsymbol{x})$. Because of $\mathcal{A}_t(\boldsymbol{x}) \subseteq \mathcal{A}_t(\boldsymbol{d})$, which follows from $\mathcal{F}(\boldsymbol{x}) \subseteq \mathcal{F}(\boldsymbol{d})$, $(\boldsymbol{x}_t)^{-1}(B)$ is contained in $\mathcal{A}_t(\boldsymbol{d})$, Hence, $\boldsymbol{x}$ is measurable with respect to $\mathcal{F}(\boldsymbol{d})$. #

If the number of random events is finite, then the restriction on $\boldsymbol{x}$ to be nonanticipative with respect to $\boldsymbol{d}$ is equivalent to the fulfillment of certain information constraints.

Corollary 1.1 (Information constraints) Assume, Ω contains a finite number of elements.
$\Longrightarrow$ The following statements are equivalent:

- $\boldsymbol{x}$ is nonanticipative with respect to $\mathcal{F}(\boldsymbol{d})$,
- process $\boldsymbol{x}$ fulfills:

$$\forall t = 1, \ldots, T \,:\, \sum_{\omega \in \Omega} H_t^{\omega} x_t(\omega) = 0\,, \tag{1.20}$$

where H_t^ω are suitable matrices, constructed from the filtration $\mathcal{F}(\boldsymbol{d})$, which are known beforehand. The entries of these matrices guarantee:

$$\begin{aligned}&\forall \omega, \nu \in \Omega,\ t = 1, \ldots, T\ : \\ &\quad \big(\forall \tau = 1, \ldots, t, d_\tau(\omega) = d_\tau(\nu)\big) \Rightarrow x_t(\omega) - x_t(\nu) = 0\ .\end{aligned} \tag{1.21}$$

The formula (1.21) generates such matrices H_t^ω. A new row is added when the values $d_\tau(\omega)$ and $d_\tau(\nu)$ do not differ for certain time periods and random events ω and ν. Generally, these matrices contain linear dependent rows. These the nonanticipativity ensuring constraints are called *information constraints*.

1.4 Scenario Tree

The stochasticity is often modeled by a stochastic process as defined in 1.1. If both, the number of time periods and the number of scenarios, are finite, then the process can be described by a finite number of values. In this case it is possible to compute expectations of functions numerically. Therefore, in this paper it is assumed:

$$\forall t = 1 \ldots T\ :\ \# x_t(\Omega) < \infty\ . \tag{1.22}$$

Since $\mathcal{A}_t$ is generated by the values of $\boldsymbol{x}_\tau$ with $\tau = 1, \ldots, t$, this σ-field contains a finite number of sets, i.e.:

$$\forall t = 1 \ldots T\ :\ \# \mathcal{A}_t < \infty\ . \tag{1.23}$$

If a function is measurable with respect to a finite σ-field, then the function is constant on sets of a certain partition that is defined by the following equivalence relation $\sim_t$:

$$\nu \sim_t \omega \Leftrightarrow \forall s = 1, \ldots, t\ :\ x_s(\nu) = x_s(\omega) \tag{1.24}$$

Let $[\omega]_t$ denote the equivalence class of scenario ω at time t:

$$[\omega]_t := \{\upsilon \in \Omega \mid \upsilon \sim_t \omega\}\ . \tag{1.25}$$

Finite σ-fields and partitions are closely related terms, since the power set and the σ-field (both generated by the sets of the partition) are the same:

$$2^{(\Omega_{/\sim_t})} = \mathcal{A}_t. \tag{1.26}$$

At the first time period, it is impossible to differentiate between the realizations of the random variable, therefore all elements of Ω are in the same set, i.e. at time $t = 1$, the partition consists of only one set. That means $\Omega_{/\sim_1} = \{\Omega\}$, therefore the following holds:

$$\mathcal{A}_1 = \{\emptyset, \Omega\}. \tag{1.27}$$

Under the assumptions given above, a scenario tree can be defined as follows.

Definition 1.5 (Scenario Tree) A scenario tree is the directed graph $G = (V, E)$:

$$\begin{aligned} V &\subseteq 2^{\Omega} \times I\!N \\ V &= \{([\omega]_t, t) \mid \omega \in \Omega, t = 1, \ldots, T\} \\ E &\subseteq V \times V \\ (([\omega]_s, s), ([v]_t, t)) \in E &\Leftrightarrow [v]_t \subseteq [\omega]_s \ \wedge \ t = s + 1 \\ [\omega]_1 &= \Omega. \end{aligned}$$

This definition combines the graph-theoretical with the stochastic description. The definition is not based on the values of the random variable. The first part of the nodes contains the scenariobundle, while the second part denotes the corresponding time period. Therefore, it is simple to switch from one description to the other. This simplifies the description of algorithms, theorems and proofs.

In Figure 1.1 an example of a scenario tree is shown, where the information about the process decreases in time, leading to more and more branches. The scenarios of this example do not differ during the first time periods, which means that the forecast is exact for these time periods. Then uncertainty leads to branches that may occur at each time period. As shown in Figure 1.1 nodes may have an arbitrary finite number (but greater than zero) of children.

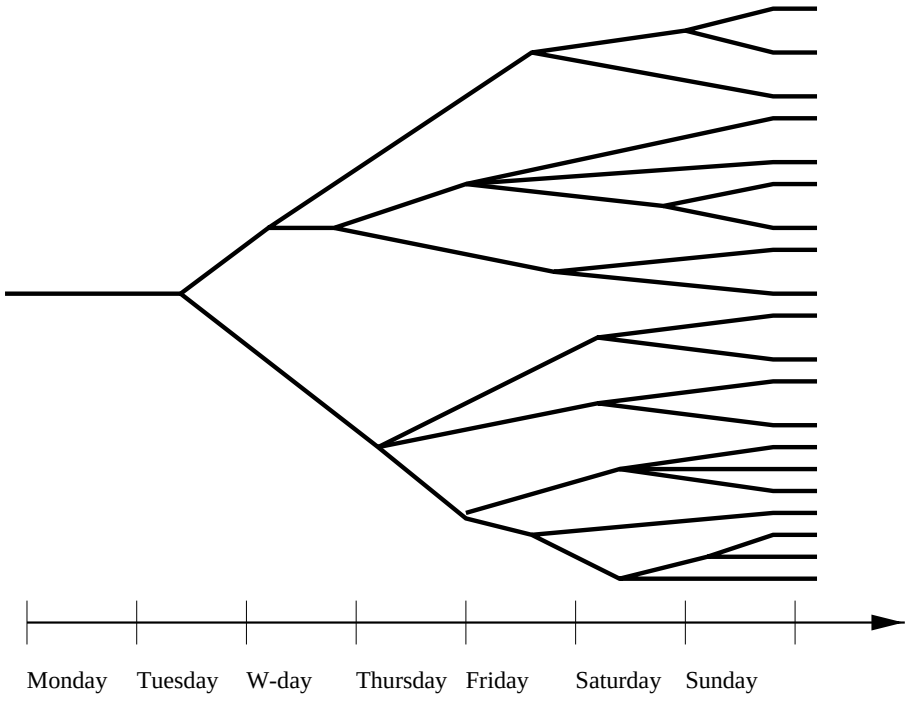

Figure 1.1: Example of a scenario tree

Now, a real-valued stochastic process $\boldsymbol{x}$ measurable with respect to the filtration $\mathcal{F}$ can be considered as a mapping $\tilde{\boldsymbol{x}} : V \to \mathbb{R}$, where V is the set of nodes of the scenario tree assigned to $\mathcal{F}$.

No restrictions are imposed on mappings $\tilde{\boldsymbol{y}}$, if $\boldsymbol{y}$ has to be only measurable with respect to $\mathcal{F}$. However, if $\boldsymbol{x}$ generates $\mathcal{F}$ (and the corresponding scenario tree), then the following corollary holds.

Corollary 1.2 Let ω and ν denote two scenarios leading to common values for the stochastic process $\boldsymbol{x}$ until $t-1$, and the scenario tree (V, E) corresponds to the filtration $\mathcal{F}$ that was generated by $\boldsymbol{x}$.
$\Longrightarrow$ Then the following equivalence holds:

$$([\omega]_t, t) = ([\nu]_t, t) \Longleftrightarrow x_t(\omega) = x_t(\nu) \tag{1.28}$$

Corollary 1.2 allows the alternative description of a scenario by the realizations $x_1(\omega), \ldots, x_T(\omega)$. However, this formulation is not suitable for the implementation.

The presented formulation and the mapping $V \to \mathbb{R}$ constitute the basis for the implementation of algorithms that use the implicit formulation of nonanticipativity.

Usually, the generating stochastic process describes the information, that is available to the decision maker. The decision process can be described by a multi-stage stochastic linear program that either uses the explicit or the implicit formulation of nonanticipativity.

1.5 Multi-Stage Stochastic Linear Programs

The two-stage model can be generalized to dynamic optimization problems extending over a finite, discrete time horizon. Multi-stage stochastic linear programs are used to describe problems a decision maker is faced with, if a sequence of decisions has to be made under the uncertainty of future realizations of the random entities. The goal is to pick the "best choice" among a set in such a way, that there always exists an opportunity to correct the decisions later, and that the costs are minimal in average. There are many different formulations of multi-stage stochastic linear programs known in literature (cf. [Dup92, KW94, BL97]). The following model is chosen to serve as a base for the algorithms described later. Although the bold notation for stochastic variables was introduced, the random event (ω) is mentioned in order to emphasize the stochasticity when it was suitable.

Problem 1.1 (Multi-Stage Stochastic Linear Program)

$$\min_{\boldsymbol{x}} \mathbb{E}^{\omega} \sum_{t=0}^{T} c_t^T x_t(\omega) \tag{1.29}$$

subject to:

$$Ax_0 = b_0 \tag{1.30}$$

$$\forall \omega \in \Omega,\, t = 1, \ldots, T \,:\, T_t(\omega)x_{t-1}(\omega) + W x_t(\omega) = b_t(\omega) \tag{1.31}$$

$$\forall \omega \in \Omega,\, t = 0, \ldots, T \;:\; l_t \leq x_t(\omega) \leq u_t \tag{1.32}$$

$$\mathcal{F}(\boldsymbol{x}) \subseteq \mathcal{F}(\boldsymbol{T}, \boldsymbol{b})\,. \tag{1.33}$$

Equation (1.31) expresses the dynamic structure of the problem. The last equations (1.33) are the nonanticipativity constraints.

There are two opportunities to formulate expression (1.33) suitable for the numerical treatment. The first version uses equations like (1.21) while the second is an implicit version, where variables with equal values share the same instance.

1.5.1 Deterministic LP-equivalent

The deterministic LP-equivalent uses the explicit formulation of the nonanticipativity constraints. In order to formulate these constraints we have to assume, that the random variables have a discrete distribution with a finite number of atoms.

Problem 1.2 (Deterministic LP-Equivalent) The deterministic LP-equivalent is a reformulation of problem 1.1, where the expectation is replaced by a weighted sum and all stochastic variables are replaced by deterministic ones with one instance for each realization. Then, the problem reads:

$$\min_x \sum_{k=1}^{K} \pi^k \sum_{t=0}^{T} c_t^T x_t^k \tag{1.34}$$

subject to:

$$\forall k = 1, \ldots, K \,:\, Ax_0^k = b_0 \tag{1.35}$$

$$\forall k = 1, \ldots, K,\, t = 1, \ldots, T \,:\, T_t^k x_{t-1}^k + W x_t^k = b_t^k \tag{1.36}$$

$$\forall k = 1, \ldots, K,\, \forall t = 0, \ldots, T \;:\; l_t \leq x_t^k \leq u_t \tag{1.37}$$

$$\forall t = 0, \ldots, T \;:\; \sum_{k=1}^{K} H_t^k x_t^k = 0, \tag{1.38}$$

where π^k is the probability of scenario k. In this formulation stochastic processes are described by separate variables for each combination of time period and scenario.

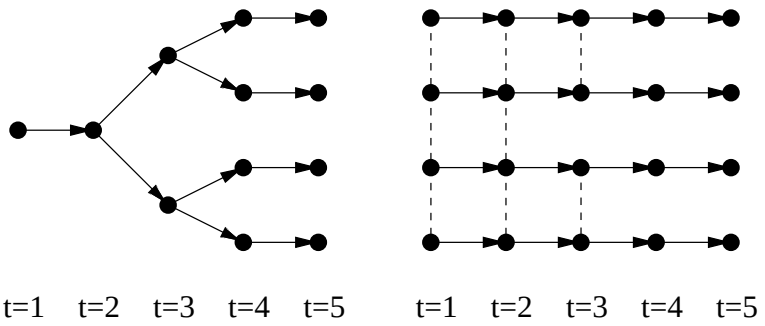

Figure 1.2: Scenario tree and deterministic LP-equivalent

The last constraints (1.38) reflect the condition under which the decision maker has to work, i.e. all decisions base on information obtained up to and including this time period.

Figure 1.2 shows a small 4-scenario example. On the left-hand side the scenario tree is shown. The decision process has to be nonanticipative with respect to the filtration which corresponds to that tree. The dark circles on the right-hand side show the decision variables used in the deterministic LP-equivalent. Since the variables have instances for each scenario, they could attain arbitrary values. However, nonanticipativity means that there should be only one value for all scenarios at the first time period. The H_t^k-matrices contain that information, i.e. that certain variables should be equal. This is denoted by broken lines in figure 1.2.

There exist different ways to model the nonanticipativity by equations. The first version models all equations for each pair of variables. For the first time period, it reads:

$$x_1^1 = x_1^2,\ x_1^1 = x_1^3,\ x_1^1 = x_1^4, \tag{1.39}$$
$$x_1^2 = x_1^3,\ x_1^2 = x_1^4,\ x_1^3 = x_1^4. \tag{1.40}$$

The second uses a subset of the first version, which corresponds to a circle:

$$x_1^1 = x_1^2,\ x_1^2 = x_1^3, \tag{1.41}$$
$$x_1^3 = x_1^4,\ x_1^4 = x_1^1. \tag{1.42}$$

The third version comes directly from the measurability of the decision process. The process $\boldsymbol{x}$ fulfills $\boldsymbol{x}_t = \mathbb{E}(\boldsymbol{x}_t|\mathcal{A}_t)$, if and only if $\boldsymbol{x}$ is nonanticipative with respect to the filtration $\mathcal{F} = \{\mathcal{A}_t\}_{t=1}^T$. For a finite number of scenarios, this leads[3] to:

$$x_t^k = \frac{1}{n_k} \sum_{\kappa \in K(k)} x_t^\kappa\,, \tag{1.43}$$

[3]assuming that the scenarios have equal probabilities

where $K(k)$ and n_k denote the conditional expectation and the number off affected scenarios, respectively. In our example this reads:

$$\begin{pmatrix} +\frac{3}{4} & -\frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} \\ -\frac{1}{4} & +\frac{3}{4} & -\frac{1}{4} & -\frac{1}{4} \\ -\frac{1}{4} & -\frac{1}{4} & +\frac{3}{4} & -\frac{1}{4} \\ -\frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} & +\frac{3}{4} \end{pmatrix} \begin{pmatrix} x_1^1 \\ x_1^2 \\ x_1^3 \\ x_1^4 \end{pmatrix} = 0, \tag{1.44}$$

while the same reads for the third period:

$$\begin{pmatrix} +\frac{1}{2} & -\frac{1}{2} & & \\ -\frac{1}{2} & +\frac{1}{2} & & \\ & & +\frac{1}{2} & -\frac{1}{2} \\ & & -\frac{1}{2} & +\frac{1}{2} \end{pmatrix} \begin{pmatrix} x_3^1 \\ x_3^2 \\ x_3^3 \\ x_3^4 \end{pmatrix} = 0, \tag{1.45}$$

These systems of equations are linear dependent. Since Problem 1.2 did not contain any integrality constraints, superfluous equations can be omitted.

1.5.2 Deterministic Graph-Equivalent

In Problem 1.2 the stochastic decision variables were replaced by deterministic instances for each realization of the random variables. The additional constraints (1.38) formulate the information structure explicitly. They ensure that certain variables have equal values. Another way to formulate the nonanticipativity constraints is the use of just one instance for those variables that should be identical in different scenarios. The constraint (1.38) is omitted and the information structure is hidden in the set of variables. Therefore, the nonanticipativity constraints are implicitly contained in the setup of the problem. The resulting problem is also known as the network formulation[4]. For a convenient formulation, the notation of a scenario tree is used next.

Problem 1.3 (Scenario tree formulation) Let V denote the set of nodes of the scenario tree. Note, that the nodes in V are pairs $k = ([\omega]_t, t)$. The problem in graph notation reads:

$$\min_x \sum_{k \in V} P([\omega]_t)\, c_k^T x_k \tag{1.46}$$

subject to:

$$A x_{(\Omega,0)} = b_0 \tag{1.47}$$

$$\forall \omega \in \Omega,\ t = 1, \ldots, T:\ T_{([\omega]_t,t)} x_{([\omega]_{t-1},t-1)} + W x_{([\omega]_t,t)} = b_{([\omega]_t,t)} \tag{1.48}$$

[4]The term "network formulation" is misleading, since undirected networks may contain a circle, which is never the case for a scenario tree.

$$\forall \omega \in \Omega,\, t = 1, \ldots, T \,:\, l_t \leq x_{([\omega]_t, t)} \leq u_t \,, \tag{1.49}$$

where $P([\omega]_t)$ denotes the probability of the node $k = ([\omega]_t, t)$, i.e. the probability of the set $[\omega]_t$, while $(\Omega, 0)$ denotes the root of the scenario tree.

These problem formulations are used in different solution methods. Each method uses a certain advantage of the corresponding formulation.

1.6 Decomposition Schemes

Programs describing real-world-problems with uncertainties easily reach dimensions that prevent the straightforward solution by available standard software on workstations. Quite often exchanging the workstation with a number crunching machine is not an option, therefore the problem is often splitted into a bunch of subproblems of lower complexity. These problems are solvable by standard software on workstations. Often, these problems show a special structure, which makes it possible to develop adapted methods. Moreover, the aim of decomposition methods is to transform a difficult problem into a number of problems with existing solution methods.

Two-stage stochastic linear programs are solvable by a stochastic quasigradient method (cf. [EW88, EG92]). This iteration method generates a sequence of first-stage variables by solving deterministic problems that correspond to scenarios. These scenarios are either sampled out of a bunch of scenarios or the result of a discrete approximation of a continuous distribution function. Since the steplength gets smaller during the iterations, the resulting first-stage variables converge to the first-stage part of the optimal solution. Since the deterministic problems are solved one after another, the number of scenarios is not limiting factor. However, the number of scenarios has an impact on the computation time, which restricts the applicability.

Stochastic branch-and-bound methods can also deal with integrality constraints, see [NPR98]. The main idea is similar to the branch-and-bound scheme save for the bounds, which are replaced by stochastic estimates.

Primal decomposition methods make use of the hierarchical structure of the problem. At each level of the problem there are some subproblems to be evaluated in order to find the solution of that level. Primal decomposition consists in replacing these subproblems by some approximations. During the run of the decomposition method these approximations get updated. A representative for these methods is the nested Benders decomposition method which is an extension of the L-shaped method for the multi-stage linear case (see Section 1.7 or [Bir88]).

In difference to that, the dual decomposition methods relax certain constraints. Then, a dual problem, the so called master problem, has to be solved. This problem comprises simple structured problems as subproblems. The first

group of dual methods takes the information constraints as a subject for the relaxation, while the second groups relaxes some coupling constraints.

If the problem is modeled scenario-wise as described in Section 1.5.1, and if the information constraints (1.38) are subject to the relaxation, then the decomposition method is called scenario decomposition (cf. [RW91, HS91, HS96]) or just dual decomposition. Then, the dual objective function reads:

$$\max_{\lambda} \min_{x} \sum_{k=1}^{K} \sum_{t=0}^{T} (\pi^k c_t^T + \lambda_t H_t^k) x_t^k. \tag{1.50}$$

Remember that the matrices H_t^k may contain a large number of rows. Now, the inner minimization problem can be carried out for each scenario (here k) separately. Generally, the optimization of the dual function does not lead to admissible scenario solutions, i.e. solutions that meet the nonanticipativity constraints.

In case of linear convex models an augmented version exists that is called progressive hedging. An additional proximal term should ensure, that the solution of the inner problem (x) does not change much, if the outer variable (λ) slightly changes. Moreover, the nonanticipativity is enforced by that term. With the proximal term the objective functions reads:

$$\max_{\lambda} \min_{x} \left\{ \sum_{k=1}^{K} \sum_{t=0}^{T} (\pi^k c_t^T + \lambda_t H_t^k) x_t^k \right\} + \sigma \underbrace{\sum_{t=0}^{T} \left\| \sum_{k=1}^{K} H_t^k x_t^k \right\|^2}_{\text{proximal term}} \tag{1.51}$$

The proximal term guarantees that the solutions of the subproblems converge (cf. [RW91]) and finally fulfill the nonanticipativity constraints.

However, the presence of integrality constraints makes it necessary to use advanced methods. Due to the non-convexity of the underlying mixed-integer stochastic program, the decomposition methods should be followed by certain global optimization techniques (branch-and-bound, heuristics etc.) in order to obtain nonanticipative solutions. In [LW96, TBL96, TKW97] the scenario decomposition by splitting methods is combined with suitable heuristics, while the authors of [CS98, Car98, CS99] use branch-and-bound as a heuristics.

Another dual decomposition method relaxes constraints coupling groups of variables. The decomposed subproblems are still stochastic. But their structure is suitable for efficient methods. This "stochastic Lagrangian relaxation" is the topic of Section 2.2.

1.7 Nested Benders Decomposition

This section describes the algorithm that is used in the computer code MSLiP (cf. [Gas90]). The problem is decomposed into subproblems

according to the stages. These subproblems are solved repeatedly using techniques as developed in [Ben62, Bir88, BL88, VW69].

The Nested Benders Decomposition method can be used to solve the multi-stage stochastic linear program as described in Section 1.5.

$$\min_x \mathbb{E}^\omega \sum_{t=0}^{T} c_t^T x_t^\omega \tag{1.52}$$

subject to:

$$A_0 x_0 = b_0 \tag{1.53}$$

$$\forall t = 1, \ldots, T : T_t^\omega x_{t-1}^\omega + W x_t^\omega = b_t^\omega \tag{1.54}$$

$$\forall t = 1, \ldots, T : l_t \leq x_t^\omega \leq u_t, \tag{1.55}$$

where x_t^ω is measurable with respect to the filtration generated by the random vectors b_t^ω and the matrices T_t^ω.

In the above formulation, decision variables $x_0, \ldots, x_{t-2}, x_{t-1}$ do not explicitly contribute to the restrictions on x_{t+1}. This Markovian structure of the constraints permits the application of decomposition methods, especially the separation of stages corresponding to different time periods t. The first step of the method is the recursive formulation of the problem, i.e.:

$$\min_{x_0} c_0^T x_0 + \mathbb{E}^\omega Q_{([\omega]_1,1)}(x_0) \tag{1.56}$$

subject to:

$$A_0 x_0 = b_0 \, , \; l_0 \leq x_0 \leq u_0, \tag{1.57}$$

where $Q_{([\omega]_t,t)}(x_{t-1})$ is the so called "recourse function" of stage t. This function is convex and piecewise linear and defined as:

$$Q_{([\omega]_t,t)}(x_{t-1}) := \min_{x_t} c_t^T x_t + \mathbb{E}^\nu \left\{ Q_{([\nu]_{t+1},t+1)}(x_t) | \omega \right\} \tag{1.58}$$

subject to:

$$T_t^\omega x_{t-1}^\omega + W x_t^\omega = b_t^\omega \tag{1.59}$$

$$l_t \leq x_t^\omega \leq u_t, \tag{1.60}$$

except for $t = T + 1$ where $Q_{(.,T+1)}(.) \equiv 0$ is defined for simplicity. The only linkage of consecutive time periods results from (1.54). In the recursive formulation this linkage is hidden in the recourse function $Q_{([\omega]_t,t)}(x_{t-1})$. Sometimes, there does not exist a solution $x_t, x_{t+1}, \ldots, x_T$ for a given point x_{t-1}. Then, and per definition, the recourse function has the value $+\infty$. This depends on the *technology*-matrix T_t^ω and *recourse*-matrix W.

If the recourse matrix W has the form $(E, -E)$, where E stands for the identity, then the problem has a *simple recourse* function. Furthermore, for all values for x_{t-1} there exists a feasible solution for the later time periods — and the functions $Q_{([.]_t,t)}(.)$ are proper functions $(\forall x, \omega \ : \ Q_{([\omega]_t,t)}(x) < \infty)$.

If the problem has the following structure, i.e.:

$$\forall y, \exists x, \ x \geq 0 \ : \quad y = Wx \ , \tag{1.61}$$

then the problem has "complete recourse". As the name says, there exists a recourse for all first stage solution, but in comparison to the simple recourse problem it has a more complicated structure.

The case is called a "relatively complete recourse problem" (cf. [RW78, HS96]), if the system $T_t^\omega x_{t-1}^\omega + Wx = b_t^\omega, \ x \geq 0$ has a solution for all ω and x_{t-1}^ω.

The next step of the Nested Benders decomposition method is the repeated solution of subproblems that arise in the recursive formulation, where $Q_{([\omega]_t,t)}(.)$ is replaced by an approximation. With the approximation $\hat{Q}_{([\omega]_1,1)}(x_0)$ the master problem:

$$\min_{x_0} c_0^T x_0 + \mathbb{E}^\omega \hat{Q}_{([\omega]_1,1)}(x_0) \tag{1.62}$$

subject to:

$$A_0 x_0 = b_0 \ , \ l_0 \leq x_0 \leq u_0, \tag{1.63}$$

is a problem of lower dimension than the problem (1.52)–(1.55).

For the numerical treatment the constraints induced by $\hat{Q}_{([\omega]_t,t)}(x_{t-1}) < \infty$ are considered separately. These constraints are replaced by $x_{t-1} \in M_{([\omega]_{t-1},t-1)}$. Since the computation of M_k is as difficult as solving the problem (1.56)–(1.57) itself, these sets M_k are approximated, too.

The polyhedral approximations at the node $k = ([\omega]_t, t)$ in iteration l are defined as:

$$\tilde{M}_k^l := \left\{ x \mid \forall i \in I_k^l \ \ (m_k^i)^T x \leq n_k^i \right\} \tag{1.64}$$

$$\tilde{Q}_k^l(x) := \max_{j \in J_k^l} (g_k^j)^T x + h_k^j. \tag{1.65}$$

With these approximations the master problem and the subproblems read:

Problem 1.4 (Master Problem) The master problem in iteration $l+1$ with the approximated recourse function and the approximated set of feasible points reads:

$$\min_{x_0} c_0^T x_0 + \mathbb{E}^\omega \tilde{Q}^l_{([\omega]_1,1)}(x_0) \tag{1.66}$$

subject to:

$$A_0 x_0 = b_0 \ , \ l_0 \leq x_0 \leq u_0 \ , \ x_0 \in \tilde{M}^l_{([\omega]_0,0)}. \tag{1.67}$$

Note, that the set of feasible points $\tilde{M}^l_{([\omega]_0,0)}$ does not depend on ω, because of $[\omega]_0 \equiv \Omega$.

Problem 1.5 (Subproblem) For each time period the subproblem at iteration $l+1$ reads:

$$\min_x c_t^T x + \mathbb{E}^\nu \left\{ \tilde{Q}^l_{([\nu]_{t+1},t+1)}(x) | \omega \right\} \tag{1.68}$$

$$T_t^\omega x_{t-1}^\omega + Wx = b_t^\omega \tag{1.69}$$
$$l_t \leq x \leq u_t, \tag{1.70}$$
$$x \in \tilde{M}^l_{([\omega]_t,t)}, \tag{1.71}$$

except for $t = T$ where $\tilde{Q}^l_{([\nu]_{T+1},T+1)}(.) \equiv 0$ is defined for simplicity.

An advantage for the numerical treatment is the similarity of problem 1.4 and 1.5. Hence, a solution method for problem 1.5 will also solve problem 1.4. Hence, the attention is now paid to the LP-version of problem 1.5.

Problem 1.6 (LP-Problem) The problem for time period t, scenario ω, and iteration $l+1$ reads:

$$v^{l+1}_{([\omega]_t,t)} = \min_{x,\phi_k} c^T_{([\omega]_t,t)} x + \sum_{\substack{k=([\nu]_{t+1},t+1),\\ \nu \in [\omega]_t}} \phi_k P([\nu]_{t+1}) \tag{1.72}$$

$$Wx = b_t^\omega - T_t^\omega x_{t-1}^\omega \tag{1.73}$$
$$l_t \leq x \leq u_t, \tag{1.74}$$

$$k = ([\omega]_t, t), \forall i \in I_k^l \quad : \quad (m_k^i)^T x \leq n_k^i \tag{1.75}$$
$$\forall \nu \in [\omega]_t, k = ([\nu]_{t+1}, t+1), \forall j \in J_k^l \quad : \quad \phi_k - (g_k^j)^T x \geq h_k^j, \tag{1.76}$$

$P([\nu]_{t+1})$ is the probability of the node $([\nu]_{t+1}, t+1)$. During the first few iterations J_k^l is empty, therefore ϕ_k has to be removed and the remaining problem has to be solved instead.

In order to get updates for the lower polyhedral approximations the dual of problem 1.6 is considered. The dual variables provide enough information for new cutting planes.

Problem 1.7 (LP-Dual)

$$\hat{v}^{l+1}_{([\omega]_t,t)} = \max_{\kappa,\lambda,\mu} \Bigg\{ \quad \kappa_1^T (b_t^\omega - T_t^\omega x_{t-1}^\omega) + \kappa_2^T l_t - \kappa_3^T u_t \tag{1.77}$$

$$- \sum_{i \in I^l_{([\omega]_t,t)}} \lambda_i n^i_{([\omega]_t,t)} + \sum_{\substack{\nu \in [\omega]_t,\\ k=([\nu]_{t+1},t+1)}} \sum_{j \in J_k^l} \mu_{j,k} h_k^j \Bigg\} \tag{1.78}$$

subject to:

$$\kappa_1 W + \kappa_2 - \kappa_3 - \sum_{i \in I^l_{([\omega]_t,t)}} \lambda_i m^i_{([\omega]_t,t)} - \sum_{\substack{\nu \in [\omega]_t, \\ k=([\nu]_{t+1},t+1)}} \sum_{j \in J^l_k} \mu_{j,k} g^j_k = c_{([\omega]_t,t)} \tag{1.79}$$

$$\forall \nu \in [\omega]_t,\ k = ([\nu]_{t+1}, t+1)\ :\ \sum_{j \in J^l_k} \mu_{j,k} = P([\nu]_{t+1}), \tag{1.80}$$

and $\kappa_2,\ \kappa_3,\ \lambda_i,\ \mu_{j,k} \geq 0$.

There exists always a feasible point for these dual problems, since $\mu_{j,k} = \frac{P([\nu]_{t+1})}{\#J^l_k}$ fulfills (1.80) and with $\kappa_1, \lambda_i = 0$ the variables κ_2 and κ_3 can appropriately be chosen in order to fulfill (1.79).

If the dual problem is unbounded, then there exists a point $(\hat{\kappa}, \hat{\lambda}, \hat{\mu})$, which is feasible with respect to (1.79 and 1.80), and a direction $(\tilde{\kappa}, \tilde{\lambda}, 0)$ such, that all points $(\hat{\kappa}, \hat{\lambda}, \hat{\mu}) + \sigma(\tilde{\kappa}, \tilde{\lambda}, 0)$ are feasible for all $\sigma \geq 0$, and the direction satisfies:

$$\tilde{\kappa}_1^T (b^\omega_t - T^\omega_t x^\omega_{t-1}) + \tilde{\kappa}_2^T l_t - \tilde{\kappa}_3^T u_t - \sum_{\dots} \tilde{\lambda}_i n^i_{([\omega]_t,t)} > 0\,. \tag{1.81}$$

Since the fulfillment of this inequality implies the unboundness of the dual problem, the primal problem is infeasible. Hence, the provided decision x_{t-1} has no recourse. In order to avoid such points, the following inequality is added to $\tilde{M}_{([\omega]_{t-1},t-1)}$:

$$\tilde{\kappa}_1^T T^\omega_t x \leq \tilde{\kappa}_1^T b^\omega_t + \tilde{\kappa}_2^T l_t - \tilde{\kappa}_3^T u_t - \sum_{i \in I^l_{([\omega]_t,t)}} \tilde{\lambda}_i n^i_{([\omega]_t,t)}\,, \tag{1.82}$$

with other words, this "feasibility cut" reads:

$$m^{i_0}_k := \tilde{\kappa}_1^T T^\omega_t, \tag{1.83}$$

$$n^{i_0}_k := \tilde{\kappa}_1^T b^\omega_t + \tilde{\kappa}_2^T l_t - \tilde{\kappa}_3^T u_t - \sum_{i \in I^l_{([\omega]_t,t)}} \tilde{\lambda}_i n^i_{([\omega]_t,t)}, \tag{1.84}$$

$$I^{l+1}_k := I^l_k \cup \{i_0\} \tag{1.85}$$

with $k = ([\omega]_{t-1}, t-1)$ and i_0 is an additional index. In order to satisfy this additional restriction, the subproblem $([\omega]_{t-1}, t-1)$ has to be solved again. If the dual problem is bounded, the primal problem has a feasible point, and the optimal values of the dual problem and of the primal problem coincide. Since $\hat{v}^{l+1}_{([\omega]_t,t)}$ is a lower approximation of $Q_(([\omega]_t, t)(x)$ (a convex function), it follows:

$$Q_(([\omega]_t, t)(x) \geq \tilde{\kappa}_1^T (b^\omega_t - T^\omega_t x) + \tilde{\kappa}_2^T l_t - \tilde{\kappa}_3^T u_t \tag{1.86}$$

$$- \sum_{i \in I^l_{([\omega]_t,t)}} \tilde{\lambda}_i n^i_{([\omega]_t,t)} + \sum_{\substack{\nu \in [\omega]_t, \\ k=([\nu]_{t+1},t+1)}} \sum_{j \in J^l_k} \tilde{\mu}_{j,k} h^j_k, \tag{1.87}$$

where $(\tilde{\kappa}, \tilde{\lambda}, \tilde{\mu})$ is the optimal solution of the dual problem 1.7. This "optimality cut" reads:

$$
\begin{aligned}
g_{k_0}^{j_0} &:= \tilde{\kappa}_1^T T_t^\omega, && (1.88)\\
h_{k_0}^{j_0} &:= \tilde{\kappa}_1^T b_t^\omega + \tilde{\kappa}_2^T l_t - \tilde{\kappa}_3^T u_t && (1.89)\\
&\quad - \sum_{i \in I^l_{([\omega]_t,t)}} \tilde{\lambda}_i n^i_{([\omega]_t,t)} + \sum_{\substack{\nu \in [\omega]_t,\\ k=([\nu]_{t+1},t+1)}} \sum_{j \in J_k^l} \tilde{\mu}_{j,k} h_k^j, && (1.90)\\
J_{k_0}^{l+1} &:= J_{k_0}^l \cup \{j_0\} && (1.91)
\end{aligned}
$$

with $k_0 = ([\omega]_t, t)$ and j_0 is an additional index. However, this should be done only in case, when $\hat{v}^{l+1}_{([\omega]_t,t)} \neq \hat{v}^l_{([\omega]_t,t)}$. If these optimal values are equal, then this "optimality cut" does not provide additional information and can be omitted.

The lower approximations of $Q_{([\omega]_t,t)}(.)$ and $M_{([\omega]_t,t)}$ are updated with these cuts Then, the subproblems and the master problem are solved in a certain order until the approximations could no longer be improved. There are two ways for the problems to communicate with each other. Primal solutions are passed down from a problem $([\omega]_t, t)$ to all problems corresponding to the nodes $\{([\nu]_{t+1}, t+1) | \nu \in [\omega]_t\}$ in form of a new input vector $x_{([\nu]_{t+1},t+1)}$. Dual solutions are passed up from $([\omega]_t, t)$ to $([\omega]_{t-1}, t-1)$ in the form of cuts.

Figure 1.3 shows the program flow of the Nested Benders Decomposition Algorithm. In this flow chart the final run was omitted for simplicity — the last run is necessary to ensure that the solutions of the subproblems are the corresponding recourse actions to the previous problems.

Moreover, the choice of a sequencing protocol has a big influence on the computation time. The article [Gas90] gives an overview about several strategies. Another important fact is that many subproblems differ only in the right-hand-sides. The solution procedure for these problems have many steps in common — hence, these problems should be solved together (cf. [VW69]). In addition, updated problems can make use of the solution of the previous iteration.

1.8 Stochastic Dual Dynamic Programming

The following method got its name from the dynamic programming algorithm. Instead of discretizing the state space, the optimal value function of the dual problem of the subsequent stage is used to approximate the costs-to-go. The independence of the stochastic data of different stages leads to non-stochastic cost-to-go functions. Further, if complete recourse is assumed, then the feasibility is always guaranteed. Therefore, no feasibility cuts are needed, while optimality cuts are common in all scenarios. Thus, Stochastic Dual Dynamic Programming can be considered as an extension of Nested Benders

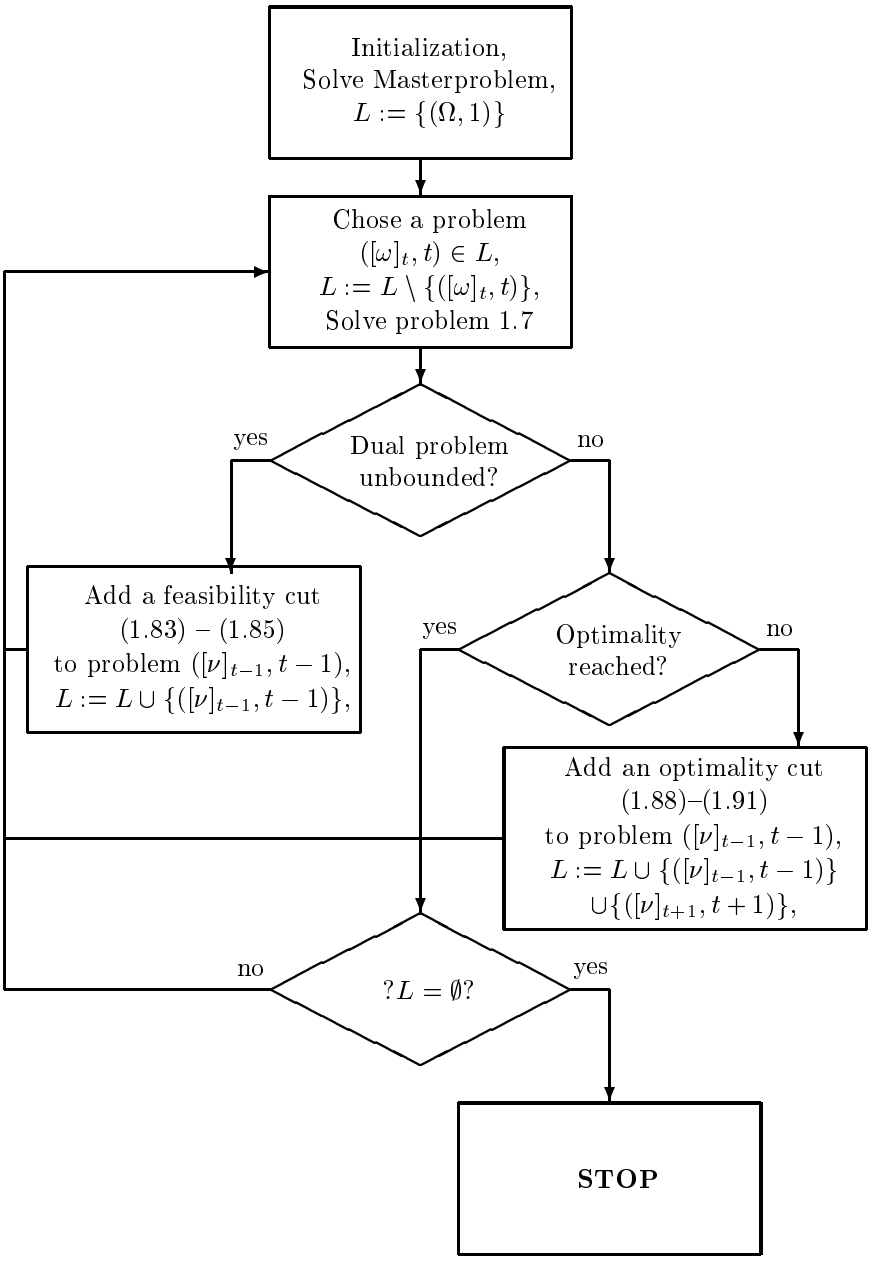

Figure 1.3: Flow chart for the Nested Benders Decomposition Algorithm

Decomposition. Since the recourse functions are non-stochastic, problems are still solvable by this method, when the equivalent number of scenarios prevents the application of other methods. In [PP91] this algorithm is developed and applied to energy planning.

Problem 1.8 (SDDP) The problem solved by Stochastic Dual Dynamic Programming reads:

$$\min_x \mathbb{E}^\omega \sum_{t=0}^{T} c_t^T x_t^\omega \tag{1.92}$$

subject to:

$$\begin{aligned} A_0 x_0 &= b_0 && (1.93)\\ \forall t = 1, \ldots, T \,:\, T_t^\omega x_{t-1}^\omega + W x_t^\omega &= b_t^\omega && (1.94) \end{aligned}$$

$$\forall t = 0, \ldots, T \,:\, l_t \leq x_t^\omega \leq u_t, \tag{1.95}$$

where x_t^ω is measurable with respect to the filtration generated by the random vectors b_t^ω and the matrices T_t^ω. Further, it is required, that

$$\begin{aligned} \forall t = 1, \ldots, T, \forall \omega \in \Omega, \forall x \in X, \exists y \;:\; & T_t^\omega x + W y = b_t^\omega && (1.96)\\ & l_{t-1} \leq x \leq u_{t-1},\, l_t \leq y \leq u_t, && (1.97) \end{aligned}$$

and that for t_1, t_2, $t_1 \neq t_2$ the random variables $(T_{t_1}^\omega, b_{t_1}^\omega)$ and $(T_{t_2}^\omega, b_{t_2}^\omega)$ are independent.

The first additional constraint ensures that there always exists a recourse action to preceding actions. The independence constraint allows the reformulation as follows:

$$\min_{x_0} c_t^T x_0 + Q_1(x_0) \tag{1.98}$$

subject to:

$$A_0 x_0 = b_0\,,\; l_0 \leq x_0 \leq u_0, \tag{1.99}$$

where Q_t is the non-stochastic recourse action of stage t. This function is convex and piecewise linear and defined as:

$$Q_t(x_{t-1}) := \mathbb{E}^\omega \min_{x_t^\omega} c_t^T x_t^\omega + Q_{t+1}(x_t^\omega) \tag{1.100}$$

subject to:

$$\begin{aligned} T_t^\omega x_{t-1} + W x_t^\omega &= b_t^\omega && (1.101)\\ l_t \leq x_t^\omega &\leq u_t, && (1.102) \end{aligned}$$

except for $t = T + 1$ where $Q_{T+1}(.) \equiv 0$ is defined for simplicity.

As in the Nested Benders Decomposition Method the cost-to-go functions $Q_t(x)$ are approximated by lower polyhedral functions $\tilde{Q}_t^l(x)$, i.e.:

$$\tilde{Q}_t^l := \max_{j \in J_t^l} (g_t^j)^T x + h_t^j \ . \tag{1.103}$$

Problem 1.9 (SDDP: LP-Problem) For a given x_{t-1} the subproblem at iteration $l + 1$ reads:

$$v_t^{l+1} := \sum_{[\omega]_t \in \Omega} \min_{x^\omega, \phi^\omega} c_t^T x^\omega + \phi^\omega P([\omega]_t) \tag{1.104}$$

subject to:

$$T_t^\omega x_{t-1} + W x^\omega = b_t^\omega \tag{1.105}$$

$$l_t \leq x^\omega \leq u_t, \tag{1.106}$$

$$\forall j \in J_t^l \ : \ \phi^\omega - (g_t^j)^T x^\omega \geq h_t^j \ . \tag{1.107}$$

The last inequality shows the lower approximation of the cost-to-go functions.

The problem 1.9 decomposes into single scenario subproblems, which can be solved separately. Under the assumptions of relatively complete recourse and the stage-wise independence of the random input data, both problems are solvable. That means, that there is no unbounded direction for the dual problem, which can be used to construct a feasibility cut — therefore there no constraint set like $\tilde{M}_t^l$ is needed. The dual solution of all scenario subproblem is used to construct an aggregated optimality cut for the non-stochastic approximation $\tilde{Q}_t^{l+1}$.

In [PP91] the authors show a version[5] for problems with a high number of scenarios. Instead of solving the subproblems for all scenarios, a Monte Carlo simulation is used to sample a subset of scenarios. The subproblems are solved for these scenarios and the results are used to get stochastic upper and lower bounds and to update the approximations.

[5]The combination of relatively complete recourse and stochastic independence of the random data for different stages allows the solution of the problem by approximated non-stochastic cost-to-go functions. These approximated recourse functions are built by cuts, which are generated by solution of certain subsets of all scenarios. This allows the treatment of problems, which are otherwise prevented by the curse of dimension.

Chapter 2

Solving the Power Scheduling Problem

This chapter focuses the solution method for a hydro-thermal power generation system. First the stochastic Lagrangian relaxation method is developed. After discussing the difficulties of a mixed-integer problem, a heuristic is presented that bases on dual parameters.

2.1 Model of a Power Generation System

The power generation system considered in this chapter comprises thermal units and pumped hydro storage plants as encountered at the German utility VEAG Vereinigte Energiewerke AG Berlin. The time horizon is 7 to 9 days as it is needed for the efficient weekly operation of hydro-thermal systems involving weekly load and pumping cycles. Due to the absence of a reliable load forecast for such a time horizon the uncertainty of the load process has to be taken into account.

This gives rise to a stochastic model of the electrical load $\{\boldsymbol{d}^t, t = 1, \ldots, T\}$ as a time-discrete stochastic process on some probability space $(\Omega, \mathcal{A}, \mathbb{P})$ reflecting that the information on the load is complete for $t = 1$, and that the uncertainty increases as t grows. For the numerical tractability it is assumed that a discrete approximation of the probability distribution is given, which means that the random variable $\boldsymbol{d}^t$ has just finitely many realizations. Let $\mathcal{F} = \{\mathcal{A}_t\}_{t=1}^T$ be the filtration generated by the load process $\boldsymbol{d}^t$, i.e. $\mathcal{A}_t := \sigma\left(\{(\boldsymbol{d}^s)^{-1}(B), s \leq t, B \in \mathcal{B}(\mathbb{R})\}\right)$, where $\mathcal{B}(\mathbb{R})$ denotes the Borel sets of $\mathbb{R}$. Due to the finiteness of the filtration a scenario tree exists.

Figure 2.1 shows an example, where the information is complete for the first time periods up to $t = 45$, and new possible observations lead to a number of additional branches. The tree structure of this load process looks like that in Figure 1.1. Here, the time scale starts on a Thursday, periods $45 - 100$ present the weekend, followed by the first 3 days of the next week.

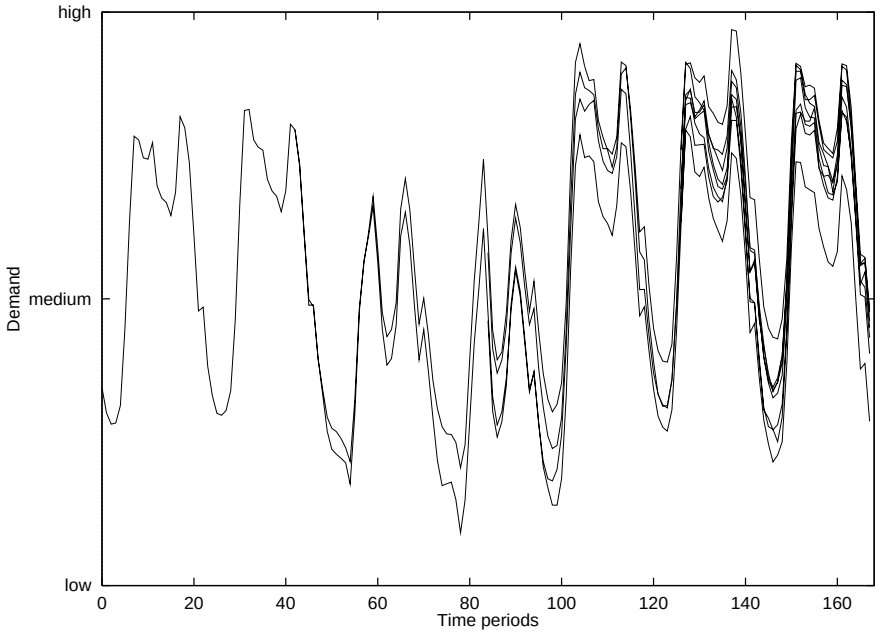

Figure 2.1: Example of a load process

Let T denote the number of time intervals obtained by discretizing the operation horizon. This discretization may be chosen uniformly (e.g. hourly) or non-uniformly. In this paper the hourly discretization was considered. In case of a non-uniformly discretization, the length of the time periods has to be taken into account. However, this does not impact the application of the presented methods to such problems.

Let I and J denote the number of thermal and pumped hydro storage units in the system. Delivery contracts are regarded as particular thermal units. The decision variables in the model correspond to the outputs of units, i.e., the electric power generated or consumed by each unit of the system, and to storage levels of the pumped hydro storage plants. The binary variables $\boldsymbol{u}_i^t \in \{0,1\}$ denote the state of unit i in time period t, thus $\boldsymbol{u}_i^t = 1$ means: unit i is online in t. Similarly $\boldsymbol{p}_i^t$ denotes the production level. The variables $\boldsymbol{s}_j^t$, $\boldsymbol{w}_j^t$ are the generation and pumping levels of the pumped hydro storage plant j during the period t, respectively. Further, $\boldsymbol{l}_j^t$ reflect the fill of the upper dam of plant j at the end of the interval t measured in terms of energy that can be generated using this storage volume. All variables mentioned above have finite upper and lower bounds representing unit limits and reservoir capacities of the generation system:

$$\forall i = 1,\ldots,I, t = 1,\ldots,T \quad : \quad \boldsymbol{u}_i^t p_i^{min} \leq \boldsymbol{p}_i^t \leq \boldsymbol{u}_i^t p_i^{max} \tag{2.1}$$

$$\forall j = 1,\ldots,J, t = 1,\ldots,T \quad : \quad 0 \leq \boldsymbol{s}_j^t \leq s_j^{max} \tag{2.2}$$

$$0 \leq \boldsymbol{w}_j^t \leq w_j^{max} \tag{2.3}$$

$$0 \leq \boldsymbol{l}_j^t \leq l_j^{max} \tag{2.4}$$

The constants $p_i^{min}, p_i^{max}, s_j^{max}, w_j^{max}$, and l_j^{max} denote the minimal/maximal outputs of the units and the maximal fill of the upper dam, respectively. The dynamics of the storage volume, which is measured in electrical energy, is modeled by the equations:

$$\forall j = 1, \ldots, J,\ \forall t = 1, \ldots, T : \boldsymbol{l}_j^t = \boldsymbol{l}_j^{t-1} - \boldsymbol{s}_j^t + \eta_j \boldsymbol{w}_j^t. \tag{2.5}$$

There exist constraints on the storage level for the first and the last time periods:

$$\forall j = 1, \ldots, J\ :\ \boldsymbol{l}_j^0 = l_j^{in},\ \boldsymbol{l}_j^T = l_j^{end}. \tag{2.6}$$

Here, l_j^{in} and l_j^{end} denote the initial and final fill of the upper dam, respectively, and η_j is the cycle efficiency of plant j. The cycle efficiency is defined as the quotient of the generation and of the pumping load that correspond to the same volume of water. The equalities (2.5) show, that there is no in- or outflows in the upper reservoirs, and hence, that the storage plants of the system operate with a constant amount of water. Together with the upper and lower bounds for $\boldsymbol{l}_j^t$ the equations (2.5) mean that certain reservoir constraints have to be maintained for all storage plants during the whole time horizon. Constraints avoiding simultaneous generation and pumping in the hydro plants are dispensable since it can be shown that such a deficiency can not occur in optimal points (cf. [GRS92]).

Further single-unit constraints are minimum up- and down-times and possible must-on/off constraints for each thermal unit. Minimum up- and down-time constraints are imposed to prevent thermal stress and high maintenance costs due to excessive unit cycling. Denoting by τ_i the minimum down-time of unit i, the corresponding constraints are described by the inequalities:

$$\forall t = 1, \ldots, T,\ \forall \tau = 1, \ldots, \min\{T - t, \tau_i - 1\}\ :\ \boldsymbol{u}_i^{t+\tau} + \boldsymbol{u}_i^{t-1} - \boldsymbol{u}_i^t \leq 1 \tag{2.7}$$

Analogous constraints can be formulated describing minimum up-times. Similar conditions are imposed to describe the initial state of the thermal units.

The next constraints are coupling power units – these are the load and reserve constraints. The first constraints are essential for the operation of the power system and express that the sum of the output powers satisfies the load demand in each time period:

$$\forall t = 1, \ldots, T\ :\ \sum_{i=1}^{I} \boldsymbol{p}_i^t + \sum_{j=1}^{J} \left(\boldsymbol{s}_j^t - \boldsymbol{w}_j^t\right) \geq \boldsymbol{d}^t. \tag{2.8}$$

For compensating unexpected events (e.g. sudden load increases or outages of units) within a specified short time period, a spinning reserve $\boldsymbol{r}^t$ is prescribed, describing the amount of additional generation available from all units synchronized on the system. The corresponding constraints are given by the following inequalities:

$$\forall t = 1, \ldots, T\ :\ \sum_{i=1}^{I} (p_i^{max} \boldsymbol{u}_i^t - \boldsymbol{p}_i^t) \geq \boldsymbol{r}^t. \tag{2.9}$$

According to the stochasticity of the electrical load, the decisions for all units are discrete-time stochastic processes as well:

$$u: \quad \Omega \times \{1,\ldots,T\} \rightarrow \{0,1\}^I \tag{2.10}$$

$$p: \quad \Omega \times \{1,\ldots,T\} \rightarrow \mathbb{R}^I \tag{2.11}$$

$$s,w,l: \quad \Omega \times \{1,\ldots,T\} \rightarrow \mathbb{R}_+^J \tag{2.12}$$

Assuming that the stochastic process $\{\boldsymbol{r}^t, t = 1,\ldots,T\}$ depends in a certain way on $\boldsymbol{d}^t$, the only informations about the random trajectory are obtained by observations of $\boldsymbol{d}^t$. These observations will only provide partial information about the development. Decisions made at time t have to anticipate the future. Without prior knowledge these decisions can depend only on the observations made so far, which is expressed by the following equations:

$$u_i^t(\omega) \;:=\; \tilde{u}_i^t(d^1(\omega), d^2(\omega), \ldots, d^t(\omega)) \tag{2.13}$$

$$\vdots \quad \vdots \quad \vdots \tag{2.14}$$

Note, that the decisions at t can react on the observations made at t. Equivalently, it is required that the stochastic decision processes are adapted to the filtration $\mathcal{F}$ generated by the stochastic load process $\boldsymbol{d}$, i.e.:

$$x^t(.) \equiv \mathbb{E}(x^t(.)|\mathcal{A}_t), \tag{2.15}$$

where x stands for $\boldsymbol{u}_i$, $\boldsymbol{p}_i$, $\boldsymbol{s}_j$, $\boldsymbol{w}_j$, and $\boldsymbol{l}_j$. The short notation for that is:

$$\mathcal{F}(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w},\boldsymbol{l}) \subseteq \mathcal{F}(\boldsymbol{d},\boldsymbol{r})\,. \tag{2.16}$$

These constraints are the so called Nonanticipativity-constraints, confer [Wet89].

> As stated above, the decisions $\boldsymbol{u}_i^t$ can react on the partial information obtained in the time periods $\tau = 1,\ldots,t$. This looks like an immediate response to observations, where in return to sudden load increases new plants were scheduled to cover these increases. However, large coal fired power plants do not have such a flexibility, because they need a preparation time for heating boilers, turbines and engines.
>
> This does not conflict with the presented model, since the proposed optimization method is applied in a learning environment, where new observation are used to update the forecast. Then, this forecast provides more information about the time ahead. At least, there is no longer uncertainty involved in the load process for the next day. This gives enough information to schedule the units for the next day. If the current time period expires, the forecast adds new time periods at the end, and the optimization starts with shifted data. Thus,

the application consists in an alternate run of forecasts and optimizations with a "rolling time horizon". After each optimization, as many units are scheduled as necessary with respect to the preparation times. Once units are committed, new optimization problems are solved with must-on and must-off constraints reflecting commitments, which have been fixed so far. In this framework, where the optimization is adapted to incoming informations, the restricted flexibility of thermal units is preserved.

The objective is to find decisions satisfying (2.1) – (2.9) such, that they minimize the expected total costs of operating the power generation system. In this model the total costs are caused by the fuel and start-up costs of the thermal units only, since hydro plants do not directly contribute to the objective function. They are used to avoid alternating operation and high-cost regions of the thermal units. Hence, their operation has an impact on the total fuel costs in the system.

Thus, the objective reads:

$$\min_{(\boldsymbol{u},\boldsymbol{p},\boldsymbol{w},\boldsymbol{s})} \mathbb{E} \sum_{i=1}^{I} \sum_{t=1}^{T} FC_i(\boldsymbol{p}_i^t, \boldsymbol{u}_i^t) + SC_i^t(\boldsymbol{u}_i), \tag{2.17}$$

where FC_i are the fuel costs of the thermal unit i during period t and SC_i^t are the start-up costs for getting the unit online in this period. The functions FC_i are piecewise linear convex, strictly monotonically increasing and of the form:

$$FC_i(p, u) = \max_{l=1,\ldots,L} \{a_{il} p + b_{il}\, u\}, \tag{2.18}$$

where a_{il} and b_{il} are fixed cost coefficients. The start-up costs $SC_i^t(\boldsymbol{u}_i)$ may vary from a maximum cold-start value to a much smaller value when the unit i is still relatively close to its operation temperature. The following description of start-up costs reflects this dependence on the down-time:

$$SC_i^t(\boldsymbol{u}_i) = \max_{\tau=0,\ldots,\tau_i^c} c_i^\tau (\boldsymbol{u}_i^t - \sum_{\kappa=1}^{\tau} \boldsymbol{u}_i^{t-\kappa}), \tag{2.19}$$

where $c_i^0 = 0$ and c_i^τ are fixed for $\tau = 1, \ldots, \tau_i^c$ increasing cost coefficients, τ_i^c is the time the unit i needs to cool down, and $c_i^{\tau_i^c}$ is its maximum cold-start costs.

Altogether, minimizing the objective function (2.17) subject to the constraints (2.1)-(2.9) leads to a cost-optimal schedule for all units of the power system during the specified time horizon. As a consequence of the strict monotonicity of the fuel costs, the cost-optimal schedule shows the following two interesting properties under normal circumstances:

- if a schedule $(\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{s}, \boldsymbol{w})$ is optimal, then the equality in load constraints (2.8) typically holds,

- the complementarity condition $\boldsymbol{s}_j^t \boldsymbol{w}_j^t = 0$ holds for all $j = 1, \ldots, J$, $t = 1, \ldots, T$,, that means generation and pumping do not occur simultaneously (if this happens it would contradict the optimality cf. [GRS92]).

In a condensed form the unit-commitment problem for a system like the VEAG-owned reads:

Problem 2.1 (VEAG-System)

$$\min_{(\boldsymbol{u},\boldsymbol{p},\boldsymbol{w},\boldsymbol{s})} \mathbb{E} \sum_{i=1}^{I} \sum_{t=1}^{T} FC_i(\boldsymbol{p}_i^t, \boldsymbol{u}_i^t) + SC_i^t(\boldsymbol{u}_i), \tag{2.20}$$

subject to:

$$\begin{aligned}
\forall i = 1, \ldots, I, t = 1, \ldots, T \quad &: \quad \boldsymbol{u}_i^t p_i^{min} \leq \boldsymbol{p}_i^t \leq \boldsymbol{u}_i^t p_i^{max} && (2.21)\\
\forall j = 1, \ldots, J, t = 1, \ldots, T \quad &: \quad 0 \leq \boldsymbol{s}_j^t \leq s_j^{max} && (2.22)\\
&\quad\; 0 \leq \boldsymbol{w}_j^t \leq w_j^{max} && (2.23)\\
&\quad\; 0 \leq \boldsymbol{l}_j^t \leq l_j^{max} && (2.24)\\
&\quad\; \boldsymbol{l}_j^t = \boldsymbol{l}_j^{t-1} - \boldsymbol{s}_j^t + \eta_j \boldsymbol{w}_j^t. && (2.25)\\
\forall j = 1, \ldots, J \quad &: \quad \boldsymbol{l}_j^0 = l_j^{in},\ \boldsymbol{l}_j^T = l_j^{end}. && (2.26)
\end{aligned}$$

$$\forall t = 1, \ldots, T \quad : \quad \sum_{i=1}^{I} \boldsymbol{p}_i^t + \sum_{j=1}^{J} \left(\boldsymbol{s}_j^t - \boldsymbol{w}_j^t \right) \geq \boldsymbol{d}^t. \tag{2.27}$$

$$\sum_{i=1}^{I} (p_i^{max} \boldsymbol{u}_i^t - \boldsymbol{p}_i^t) \geq \boldsymbol{r}^t. \tag{2.28}$$

The minimization problem represents a mixed-integer program with linear constraints, and IT binary and $(I + 2J)T$ continuous decision variables, respectively. For a typical configuration of the VEAG-owned generation system with $I = 25$ (thermal), $J = 7$ (hydro) and $T = 168$ (7 days with hourly discretization), the dimension of the model is shown in the first row of Figure 2.1. The first row contains the number of scenarios. The corresponding number of nodes of the scenario tree depends on the structure of the scenario tree. The scenario tree contains one more node for each time period the new branch starts earlier. In Table 2.1 the branching points are almost equidistantly distributed. The remaining rows show the corresponding numbers of variables, constraints, and non-zeros.

2.2 Stochastic Lagrangian Relaxation

The Lagrangian relaxation is a solution method for constrained problems like:

$$\min_x \{ f(x) | \forall i = 1, \ldots, m,\ g_i(x) \leq 0 \}\ , \tag{2.29}$$

Scenarios	Nodes	Variables		Constraints	Non-zeros
		binary	continuous		
1	168	4200	6652	13441	19657
5	462	11550	18018	36965	54059
10	756	18900	29484	60490	88462
20	1176	29400	45864	94100	137612
30	1663	41575	64857	133070	194601
50	2478	61950	96642	198290	289976
80	3696	92400	144144	295760	432512
100	4200	105000	163800	336100	491500

Table 2.1: Dimension of the mixed-integer LP depending on the number of scenarios with T=168, I=25 and J=7

where $x \in \mathbb{R}^n$, and f and g_i are convex functions. The functions g_i describe a convex set that is not known beforehand. Since unconstrained problems are easier to solve (for instance with a subgradient method), a transformation of the constrained problem into a certain unconstrained problem could lead to a solution method for such problems.

Penalty methods include the constraints into the objective function in order to get an unconstrained problem, which reads now:

$$P(\sigma): \qquad \min_x f(x) + \sigma \sum_{i=1}^{m} |g_i(x)^+|^2 , \tag{2.30}$$

with σ as a penalty parameter and $z^+ = max(0, z)$. Under reasonable assumptions, the sequence of solutions $x(\sigma)$ of the problems $P(\sigma)$ converges to a solution of problem 2.29 as σ goes to infinity. Another formulation of the penalty term leads to the Lagrangian relaxation:

$$\min_x f(x) + \sum_{i=1}^{m} \underbrace{(\sigma g_i(x)^+)}_{\lambda_i(\sigma)} g_i(x) , \tag{2.31}$$

where $\lambda_i(\sigma)$ take the part of Lagrange parameters. In case of f and g_i being smooth functions, the $\lambda_i(\sigma)$ values converge to the some values, due to the necessary optimality condition for smooth convex problems.

2.2.1 Lagrangian Relaxation

The Lagrangian relaxation approach bases on the Lagrange function:

$$L(x, \lambda) := f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) , \tag{2.32}$$

where λ_i are the so called Lagrange parameters, while f and g_i are convex function for $x \in \mathbb{R}^n \mapsto \mathbb{R}$. The problem:

$$(P) \qquad \min_x \{f(x) | \forall i = 1, \ldots, m,\ g_i(x) \leq 0\} \tag{2.33}$$

is equivalent to the unconstrained problem:

$$v_P := \inf_{x \in \mathbb{R}^n} \tilde{f}(x), \text{ with } \tilde{f}(x) := \sup_{\lambda \in \mathbb{R}^m_+} L(x, \lambda). \tag{2.34}$$

The function $\tilde{f}$ is an extended real valued function being equal to $+\infty$ on all infeasible points of the primal problem (2.33). The exchange of the "inf" and the "sup" yields the dual problem:

$$v_D := \sup_{\lambda \in \mathbb{R}^m_+} \tilde{h}(\lambda), \text{ with } \tilde{h}(\lambda) := \inf_{x \in \mathbb{R}^n} L(x, \lambda). \tag{2.35}$$

The Lagrange function $L(x, \lambda)$ is convex with respect to x and linear with respect to λ. The geometric interpretation of such a surface gave the name for the following point. A point $(\bar{x}, \bar{\lambda})$ is called *saddle point*, if

$$\forall x \in \mathbb{R}^n, \lambda \in \mathbb{R}^m_+ \ : \ L(\bar{x}, \lambda) \leq L(\bar{x}, \bar{\lambda}) \leq L(x, \bar{\lambda}) \tag{2.36}$$

holds.

Proposition 2.1 If the point $(\bar{x}, \bar{\lambda})$ is a saddle point for L, then $\bar{x}$ solves problem (2.34), while $\bar{\lambda}$ solves (2.35). Moreover, v_P equals v_D.

This situation is called strong duality. If the Slater condition, i.e.

$$v_P \in \mathbb{R} \wedge \exists\, x_0, \forall i = 1, \ldots, m \ : \ g_i(x_0) < 0\,, \tag{2.37}$$

is satisfied, then the strong duality holds. Hence, the constrained problem 2.33 is solved, if a saddle point for the corresponding Lagrange function was obtained.

2.2.2 Stochastic Lagrangian Relaxation for Dynamic Recourse Problems

The stochastic Lagrangian relaxation is slightly different to the Lagrangian relaxation presented above. Subject for the relaxation is a constraint that has to be fulfilled for μ-almost all scenarios. The approach is illustrated on the following two-stage problem.

Problem 2.2 Let $(\Omega, \mathcal{A}, \mathbb{P})$ be a probability space, $\mu \in \mathbb{P}$. The sets C_X and C_Y are compact and convex. Further, $x \in \mathbb{R}^n$ and $y \in L^\infty(\Omega, \mathcal{A}, \mu; \mathbb{R}^m)$ are the decision variables. The functions $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l \mapsto \mathbb{R}$ and $g : \mathbb{R}^m \mapsto \mathbb{R}^k$ are convex. The

stochastic data $d(\omega)$ and $h(\omega)$ are elements of $L^\infty(\Omega, \mathcal{A}, \mu; \mathbb{R}^l)$ and $L^\infty(\Omega, \mathcal{A}, \mu; \mathbb{R}^k)$, respectively. Then, the problem reads:

$$\min_{x \in X \cap C_X} \mathbb{E} \min_{y(\omega) \in Y \cap C_Y} f(x, y(\omega), d(\omega)) \tag{2.38}$$

subject to:

$$\forall \omega \in \Omega \; : \; g(y(\omega)) \leq h(\omega) \, . \tag{2.39}$$

The stochastic constraint (2.39) is subject for the relaxation. Since (2.39 is a inequality constraint with objects from L^∞, the appropriate Lagrange multipliers are elements of L^1 (cf. [RW78]), i.e. $\lambda \in L^1(\Omega, \mathcal{A}, \mu; \mathbb{R}^k)$. Using these multipliers the straightforward formulation of the corresponding Lagrange functions reads:

$$\{\mathbb{E} f(x, y(\omega), d(\omega))\} + < \lambda(.), g(y(.)) - h(.) >_\Omega \, , \tag{2.40}$$

where $< ., . >_\Omega$ denotes a suitable scalar product.

In distinction to Lagrangian relaxation within the deterministic setting, the stochastic nature of the underlying problem should be taken into account. The Lagrange parameters denote some prices for the capacities (2.39) — they should be independent of the probability of the corresponding scenario.

In Problem 2.2, the Lagrange parameters denote capacity prices at the second stage. Assume now, that the problem was solved by some method yielding optimal first-stage variables. Then, the problem with fixed first-stage variables decomposes into single scenario problems. Since these problems do not contain any stochasticity, they are solvable by Lagrangian relaxation. The Lagrange parameter for a scenario returned by Lagrangian relaxation should equal the corresponding parameter of the stochastic version. Therefore, the scalar product is defined as:

$$< \lambda(.), g(y(.)) - h(.) >_\Omega := \mathbb{E} \left\{ \lambda(\omega)^T \left(g(y(\omega)) - h(\omega) \right) \right\} \, . \tag{2.41}$$

The difference between Lagrangian relaxation of a stochastic program and the stochastic Lagrangian relaxation gets clear, if a problem with a finite number of scenarios is considered. The first one is the relaxation of the deterministic graph-equivalent, which leads to:

$$\sum_{k=1}^{K} \pi_k f_k(x, y_k) + \lambda_k (g(y_k) - h_k) \tag{2.42}$$

as the corresponding Lagrange function, while the deterministic graph-equivalent of the stochastic Lagrangian relaxation reads:

$$\sum_{k=1}^{K} \pi_k \left\{ f_k(x, y_k) + \lambda_k (g(y_k) - h_k) \right\} \, . \tag{2.43}$$

Moreover, in a two-scenario example with identical scenarios, the stochastic Lagrangian relaxation gives the same values as for the corresponding one-scenario problem — the other approach gives just half as big Lagrange parameters.

For fixed Lagrange parameters, the problem might decompose into a number of stochastic subproblems. Sometimes these subproblems are easier to solve than the originating problem. The measurability of the Lagrange parameters plays an important part.

Problem 2.3 (Decomposable Multi-stage Problem) Let $\boldsymbol{x}$ and $\boldsymbol{y}$ denote stochastic decision variables, while the input process contains the variables $\boldsymbol{T}$, $\boldsymbol{b}$ and $\boldsymbol{z}$. The problem reads:

$$\min_{\boldsymbol{x},\boldsymbol{y}} \mathbb{E} \sum_{t=1}^{T} c_t^T \boldsymbol{x}_t + d_t^T \boldsymbol{y}_t \tag{2.44}$$

subject to:

$$\begin{aligned} \forall t = 2, \ldots, T \,:\, \boldsymbol{T}_t \boldsymbol{x}_{t-1} + W \boldsymbol{x}_t &= \boldsymbol{b}_t & (2.45)\\ \forall t = 1, \ldots, T \,:\, F_t \boldsymbol{x}_t &\leq f_t & (2.46)\\ \forall t = 1, \ldots, T \,:\, G_t \boldsymbol{y}_t &\leq g_t & (2.47)\\ \forall t = 1, \ldots, T \,:\, U_t \boldsymbol{x}_t + V_t \boldsymbol{y}_t &\leq \boldsymbol{z}_t \,. & (2.48) \end{aligned}$$

Equation (2.45) expresses the dynamics of the system, while (2.48) shows, that $\boldsymbol{x}$ and $\boldsymbol{y}$ share common capacities.

The relaxation of the constraint (2.48) yields the Lagrange function:

$$L(\boldsymbol{\lambda}; \boldsymbol{x}, \boldsymbol{y}) := \mathbb{E} \sum_{t=1}^{T} c_t^T \boldsymbol{x}_t + d_t^T \boldsymbol{y}_t + \boldsymbol{\lambda}_t \left(U_t \boldsymbol{x}_t + V_t \boldsymbol{y}_t - \boldsymbol{z}_t \right) \,, \tag{2.49}$$

where $\boldsymbol{\lambda}_t$ are suitable Lagrange parameters. A stochastic Lagrange multiplier corresponds to each constraint (2.48). Since $(\boldsymbol{x}, \boldsymbol{y})$ has to be nonanticipative with respect to the filtration $\mathcal{F}(\boldsymbol{T}, \boldsymbol{b}, \boldsymbol{z})$, the term $U_t \boldsymbol{x}_t + V_t \boldsymbol{y}_t - \boldsymbol{z}_t$ is measurable with respect to $\mathcal{A}_t(\boldsymbol{T}, \boldsymbol{b}, \boldsymbol{z})$. Therefore, the appropriate Lagrange multiplier $\boldsymbol{\lambda}_t$ is an element of $L^1(\Omega, \mathcal{A}_t(\boldsymbol{T}, \boldsymbol{b}, \boldsymbol{z}), \mu; \mathbb{R}^k)$, where $\mathcal{A}_t$ denotes the σ-field of the filtration $\mathcal{F}$ assigned to time period t and k is a suitable dimension (cf. [RW76]).

This is the main difference to scenario decomposition. There, the dual parameter satisfies a different measurability. In scenario decomposition dual variables should enforce the nonanticipativity of primal variables. If $\lambda_t(\omega)$ is the dual parameter to a certain primal variable, and this primal variable corresponds to the node $([\omega]_t, t)$ of the scenario tree, then $\boldsymbol{\lambda}_t$ is measurable with respect to

$$\tilde{\mathcal{A}}_t := \sigma \left\{ [\nu]_s | \nu \in [\omega]_s, s \geq t \right\} \,. \tag{2.50}$$

These σ-fields form a structure, which is inverse to filtrations. This inverse structure can not be used to reduce the storage requirements for the scenario decomposition method. Hence, a dual variable is used for each splitted primal variable,

which leads to a dual problem of high dimension. The stochastic Lagrangian relaxation of few constraints leads to dual problems of low dimension.

The common measurability of primal and dual variables in the stochastic Lagrangian relaxation leads to subproblems, that often have well known counterparts in (deterministic) optimization.

For fixed $\boldsymbol{\lambda}$ the Problem 2.3 decomposes into a dynamic problem in $\boldsymbol{x}$, i.e.:

$$\min_{\boldsymbol{x}} \mathbb{E} \sum_{t=1}^{T} c_t^T \boldsymbol{x}_t + \boldsymbol{\lambda}_t U_t \boldsymbol{x}_t \tag{2.51}$$

subject to:

$$\forall t = 2, \ldots, T \; : \; \boldsymbol{T}_t \boldsymbol{x}_{t-1} + W \boldsymbol{x}_t \;=\; \boldsymbol{b}_t \tag{2.52}$$

$$\forall t = 1, \ldots, T \; : \; F_t \boldsymbol{x}_t \;\leq\; f_t \,, \tag{2.53}$$

and into a bunch of problems in $\boldsymbol{y}_t$, that are for each $\omega \in \Omega, t = 1, \ldots, T$:

$$\min_{y} \left\{ d_t^T y_t + \lambda_t(\omega) V_t y | G_t y_t \leq g_t \right\} . \tag{2.54}$$

Such a situation is not as artificial as it seems, because the problem models the situation, that an enterprise can use the market in order to satisfy certain demands. Then, the problems in $\boldsymbol{y}_t$ present the trading of such commodities. Since the constraint of sharing a resource/capacity was relaxed, the problem in $\boldsymbol{x}$ could decompose in subproblems, too.

2.3 Decomposition

The approach dicussed in this section consists in a "stochastic version" of the classical Lagrangian relaxation idea ([Lem92]) that is very popular in power optimization ([BLSP83, FK98, GMR+97, LR96, SF94, WW96, ZG88]). That means, that the demand and reserve constraints are subject for the stochastic Lagrangian relaxation. This relaxation of a unit commitment problem with uncertain demand leads to stochastic subproblems — their counterparts are network flow problems and dynamic programming problems including continuous variables.

Associating stochastic Lagrange multipliers with the unit coupling constraints

(2.27) and (2.28) leads to the Lagrangian L:

$$L(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w};\boldsymbol{\lambda},\boldsymbol{\mu}) = \mathbb{E}\sum_{t=1}^{T}\left\{\sum_{i=1}^{I} FC_i(\boldsymbol{p}_i^t,\boldsymbol{u}_i^t) + SC_i^t(\boldsymbol{u}_i)\right. \tag{2.55}$$

$$+\boldsymbol{\lambda}^t\left(\boldsymbol{d}^t - \sum_{i=1}^{I}\boldsymbol{p}_i^t - \sum_{j=1}^{J}(\boldsymbol{s}_j^t - \boldsymbol{w}_j^t)\right) \tag{2.56}$$

$$\left.+\boldsymbol{\mu}^t\left(\boldsymbol{r}^t - \sum_{i=1}^{I}(\boldsymbol{u}_i^t p_i^{max} - \boldsymbol{p}_i^t)\right)\right\}. \tag{2.57}$$

Using this Lagrange function the dual problem reads:

Problem 2.4 (Problem, dual to the VEAG-Problem 2.1)

$$\max_{\boldsymbol{\lambda},\boldsymbol{\mu}\geq 0}\min_{(\boldsymbol{u},\boldsymbol{p},\boldsymbol{w},\boldsymbol{s})} L(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w};\boldsymbol{\lambda},\boldsymbol{\mu}) \tag{2.58}$$

subject to:

$$\forall i=1,\ldots,I, t=1,\ldots,T \;:\; \boldsymbol{u}_i^t p_i^{min} \leq \boldsymbol{p}_i^t \leq \boldsymbol{u}_i^t p_i^{max} \tag{2.59}$$

$$\forall j=1,\ldots,J, t=1,\ldots,T \;:\; 0 \leq \boldsymbol{s}_j^t \leq s_j^{max} \tag{2.60}$$

$$0 \leq \boldsymbol{w}_j^t \leq w_j^{max} \tag{2.61}$$

$$0 \leq \boldsymbol{l}_j^t \leq l_j^{max} \tag{2.62}$$

$$\boldsymbol{l}_j^t = \boldsymbol{l}_j^{t-1} - \boldsymbol{s}_j^t + \eta_j \boldsymbol{w}_j^t. \tag{2.63}$$

$$\forall j=1,\ldots,J \;:\; \boldsymbol{l}_j^0 = l_j^{in},\ \boldsymbol{l}_j^T = l_j^{end}. \tag{2.64}$$

Due to the relaxation of the coupling constraints (2.27) and (2.28), the minimization in (2.58) decomposes into stochastic single unit subproblems and the dual function takes the form

$$D(\boldsymbol{\lambda},\boldsymbol{\mu}) = \sum_{i=1}^{I}\tilde{D}_i(\boldsymbol{\lambda},\boldsymbol{\mu}) + \sum_{j=1}^{J}\hat{D}_j(\boldsymbol{\lambda}) + \mathbb{E}\sum_{t=1}^{T}[\boldsymbol{\lambda}^t\boldsymbol{d}^t + \boldsymbol{\mu}^t\boldsymbol{r}^t], \tag{2.65}$$

where $\tilde{D}_i(\boldsymbol{\lambda},\boldsymbol{\mu})$ and $\hat{D}_j(\boldsymbol{\lambda})$ refer to the optimal values of the thermal subproblem (4.2) and the hydro storage subproblem (4.1), respectively. The dual function D is concave and non-differentiable on $\mathbb{R}^{2N}$, but of lower dimension than the primal model in problem 2.1. Hence, the approach is based on the same, but *stochastic*, ingredients as in the classical case: a solver for the non-differentiable dual, subproblem solvers, and a Lagrangian heuristics. It turns out that, with a state-of-the-art bundle method for solving the dual, efficient stochastic subproblem solvers based on a specific descent algorithm and stochastic dynamic programming, respectively, and a specific Lagrangian heuristics for determining a nearly optimal solution, this *stochastic Lagrangian relaxation* algorithm becomes efficient.

Scenarios	Nodes (N)	Dual Variables	Primal Variables
1	168	336	10852
5	462	924	29568
10	756	1512	48384
20	1176	2352	75264
30	1663	3326	106432
50	2478	4956	158592
80	3696	7392	236544
100	4200	8400	268800

Table 2.2: Dimension of the dual problem in comparison to the primal problem

2.4 Duality Gap in Mixed-Integer Programs

Mixed-integer programs show a behavior unlike the one that is known from linear programs. Linear programs are convex, and if feasible points exist for both, the primal and the dual problem, then the optimal values of the primal and the dual problem coincide. This is also known as strong duality[1]. This is no longer valid in the mixed-integer case, even if all constraints but the integrality constraints are formulated in linear form.

> The appearance of integrality constraints kicks us out of the nice framework of linear programs. The equivalent formulation is the formulation as a nested minimization. Then, the objective functions is the minimum of the objective values of a finite number[2] of ordinary linear programs. On the one hand, the objective value function is not convex, since the minimum of convex functions is no longer a convex function. On the other hand, integer variables appear in the constraints too, which results in varying domains for the linear programs. Outside the domain, the objective value of a minimization problem equals infinity by definition. The minimization of such extended real valued functions leads to a function with discontinuities.

The appearance of integrality constraints has a combinatorial effect. In order to find the optimal value, several combinations of integer variables have to be investigated. The dependence of the set of optimal points on the right hand side shows two different situations:

- Two combinations of integer variables result in almost the same optimal value. If the right hand side changes in a certain direction, then one combination replaces the other as the optimal solution. Thus, changing the right

[1] cf. page 38

[2] Assuming, that integer variables are bounded.

hand side slightly can lead to a completely different set of optimal points, even if the objective function shows some continuity.

- A slight change of the right hand side allows a combination of integer variables to correspond to a feasible linear problem. Then, the minimization over the integer variables has one more candidate to consider. Conversely, a slight change of the right hand side may destroy the feasibility of a certain linear subproblem. Hence, changing the right hand side slightly may lead to jumps of the objective function.

This combinatorial effect requires the evaluation of the objective function for a number of combinations that are feasible and nearly optimal or just nearly feasible. The high dimension of the programs prevents the solution by complete enumeration. Branch-and-bound methods run out of memory, if the number of candidates for the optimal solution is too big.

If the real optimum is not available, one might look for a reasonably good point instead, which could be obtained by a decomposition method. Dual decomposition methods relax some constraints, which are penalized by Lagrange multipliers and taken into the objective function. Now, the remaining constraints are separable, and the problem can be splitted into subproblems.

Such an approach works well in the continuous case. However in the mixed-integer case, some information linkage between the integer variables is cut. An example for such an information linkage is the case, if just one out of n integer variables appears to be nonzero in the optimal solution. Such a behavior is not even reflected by the subproblems of the dual program, because dual variables can not carry combinatorial information. This is the price, which has to be paid for the advantage of a smaller complexity of the dual problem. A small example is illustrating this situation.

Problem 2.5 (One stage, one unit) The unit has an objective function, which is affine linear for the online state and zero otherwise. The output of the unit has to be within a certain range, which depends on the state of the unit. The primal problem reads:

$$\min_{(u,p)} \left\{ ap + bu \quad | \quad p = D,\ p^{min}u \leq p \leq p^{max}u \right\}. \tag{2.66}$$

Further, $u \in \{0,1\}$, $p \in \mathbb{R}$, and D denotes the demand. A reasonable assumption are positive fixed and operational costs, i.e. $b > 0$ and $a > 0$, respectively. For values $D \in [p^{min}, p^{max}]$ the primal problem yields an optimal value $v_P := aD + b$.

Relaxing $p = D$ one gets

$$\max_{\lambda} \min \left\{ \min_{p} \left(ap + b + \lambda(D - p) \right), \lambda D \right\} \tag{2.67}$$

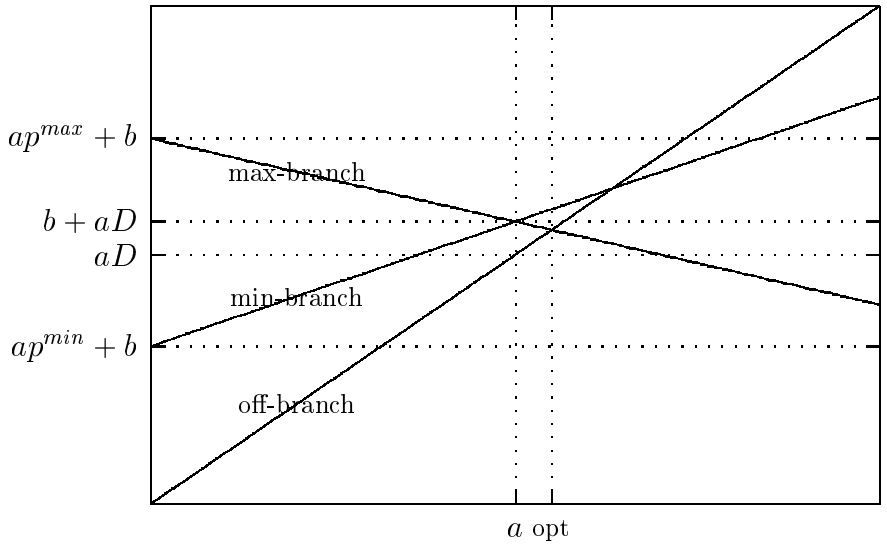

Figure 2.2: Example: one stage, one unit

as the corresponding dual problem. Hereby, the minimization with respect to u is unrolled — the two terms for the online and offline state are stated explicitly. The minimization with respect to p is one-dimensional and three cases have to be considered:

$$\min_p \left(ap + b + \lambda(D - p)\right) := \begin{cases} ap^{max} + b + \lambda(D - p^{max}) & \lambda > a \\ b + aD & \lambda = a \\ ap^{min} + b + \lambda(D - p^{min}) & \lambda < a\,. \end{cases} \tag{2.68}$$

The maximization in (2.67) is separately considered for these 3 regions, which leads to an optimal value for the dual problem of $v_D := (a + b/p^{max})D$.

Note, that there is a gap between the optimal value of the primal problem and that one of the dual problem:

$$v_P - v_D = \left(1 - \frac{1}{p^{max}} D\right) b\,. \tag{2.69}$$

Note, this dual gap vanishes as D approaches p^{max}.

Figure 2.2 explains, what happened. The dual problem lacks the information, that the unit has to be online[3]. Therefore, the intersection of the *max-branch* and the *off-branch* is chosen as the maximal value for the dual problem, while the intersection of the *max-branch* with the *min-branch* appears to be the optimum for the primal problem.

The problem 2.5 shows that a duality gap appears, if combinatorial information is omitted.

[3]Otherwise, the demand cannot be satisfied!

This example with just one unit and one time period is not as artificial as it seems, because the request of one unit being online appears in all scheduling problems.

Is is not possible to close the duality gap using just linear functions. The incorporation of constraints into the objective function by means of other than linear functions prevents the decomposition. For instance, sub-additive functions might be able to close the gap, since they can carry combinatorial information. However, the problem still has its complexity. In almost all mixed-integer problems the optimal solution shows some combinatorial aspect.

The lack of convexity in the mixed-integer case is a result of the presence of integer variables. The combinatorial aspect prevents the existence of optimal Lagrange parameters, such that strong duality holds. However, the objective value of the dual problem can be used to certify the quality of a solution that was obtained by some heuristics.

If the primal point corresponding to the optimal dual parameters as the solution of the inner minimization problem is feasible, then this primal point is optimal too, because the penalty term vanishes. This can be used to obtain upper bounds for the duality gap (cf. [Ber82]).

2.5 Lagrangian Reduction

Solving the dual problem provides Lagrange multipliers for the so called "dual information" about the relaxed constraints. In convex optimization non-zero multipliers correspond to active constraints, which are necessary for the optimization. In distinction to that, constraints with zero multipliers can be omitted.

The information about active constraints can be used to derive primal feasible points. Moreover, primal feasible points preferred by the dual information, which are feasible, are also optimal for the primal problem.

Theorem 2.1 (Optimality of Crossover Points) Let f and g be convex functions on $conv(X)$ that is a compact set. The considered primal problem reads:

$$v_P := \min \{ f(x) \mid x \in X,\ g(x) \leq 0 \} \ . \tag{2.70}$$

The variable $\tilde{\lambda}$ denotes an optimal point for the dual problem, i.e.:

$$v_D := \max_{\lambda \geq 0} \min_{x \in X} f(x) + \lambda^T g(x), \tag{2.71}$$

while $\tilde{X}$ denotes the set of dual preferred primal points, i.e.:

$$\tilde{X} := \arg\min_{x \in X} f(x) + \tilde{\lambda}^T g(x). \tag{2.72}$$

$\Longrightarrow$ The intersection $\{x | g(x) \leq 0\} \cap \tilde{X}$ comprises optimal points.

Proof: Since f and g where convex function and the set X was bounded, the weak duality holds, i.e.:

- $\exists x_0, x_0 \in X, g(x_0) \leq 0, v_P = f(x_0)$ and
- $v_D \leq v_P$.

Because $\tilde{\lambda}$ was optimal for the dual problem (2.71), it follows:

$$v_D = \min_{x \in X} f(x) + \tilde{\lambda}^T g(x)\,. \tag{2.73}$$

This values is also attained by all elements of $\tilde{X}$:

$$\forall x \in \tilde{X} \;:\; v_D = f(x) + \tilde{\lambda}^T g(x)\,. \tag{2.74}$$

The last term is less than zero for all feasible x, i.e.:

$$\forall x \in \tilde{X} \;:\; g(x) \leq 0 \Longrightarrow f(x) \leq v_D \leq v_P\;. \tag{2.75}$$

But x feasible and $f(x) \leq v_P$ means, that x is optimal. #

Even in case of convex problems, which satisfy the Slater-condition, the dual information is not sufficient for obtaining feasible points. The Lagrange method can not guarantee to return Lagrange multipliers, which lead to a non-empty intersection with the feasibility set. Or the set of dual preferred points might be too large in order to get hints about the primal solution. The following example illustrates this situation.

Problem 2.6 (Example for the Lagrangian Relaxation)

$$\min_{x_1, x_2, z} x_1^2 + x_2^2$$

s.t.:

$$x_1 + z \geq 5,\; x_2 - z \geq 3\,.$$

The optimal point for this problem is $x_1 = 4$, $x_2 = 4$, and $z = 1$.

Now, the constraints are subject to the Lagrangian relaxation, thus the resulting dual problem reads:

Problem 2.7 (Corresponding Dual Problem)

$$\max_{\lambda_1, \lambda_2 \geq 0} \min_{x_1, x_2, z} x_1^2 + x_2^2 + \lambda_1(5 - x_1 - z) + \lambda_2(3 - x_2 + z)\,.$$

After sorting the variables the (dual) objective function reads:

$$\max_{\lambda_1,\lambda_2\geq 0} \min_{x_1,x_2} x_1^2 - \lambda_1 x_1 + x_2^2 - \lambda_2 x_2 + 5\lambda_1 + 3\lambda_2 + \underbrace{\min_z (\lambda_2 - \lambda_1) z}_{=-\infty \text{ iff } \lambda_1 \neq \lambda_2} .$$

In order to get a finite value for the dual problem, the Lagrange parameters have to have equal values, otherwise the inner minimization tends to $-\infty$ due to the marked term, because z is unbounded.
Thus, the dual problem reduces to:

$$\max_{\lambda\geq 0} \min_{x_1,x_2} x_1^2 + x_2^2 - \lambda(x_1 + x_2) + 8\lambda \,, \tag{2.76}$$

which simplifies to $\max_{\lambda\geq 0} \min_x 2x^2 - 2\lambda x + 8\lambda$. The optimality conditions give $x = \frac{\lambda}{2}$ and $\frac{\delta}{\delta\lambda}(\frac{\lambda^2}{2} - \lambda^2 + 8\lambda) = 0 \Rightarrow \lambda = 8$. Therefore, $x_1 = x_2 = 4$ is obtained as a result of the dual approach. Note, the dualization lost some information about z. In order to obtain a solution for the primal problem, a z must be found such that $(4, 4, z)$ is a feasible point.

This small example shows, that only partial information about primal points can be gained from dual information. If one applies a numerical method to the dual problem, the algorithms returns an arbitrary value for z, which need not be feasible.

The decomposition method was applied to the power scheduling problem with uncertain demand in order to avoid the complexity due to the integer variables. Hence, the interest has the dependency of the schedules on the dual parameters, which present prices for the demand and reserve capacities. The question arises, which schedules are preferred, and how this preference changes with respect to perturbed prices.

Problem 2.8 (Mixed-Integer Problem) The mixed-integer problem suitable for Lagrangian reduction reads:

$$\min \{g_0(b, c) | \forall i \in I \,:\, g_i(b, c) \geq d_i\} \,, \tag{2.77}$$

where b and c denote binary variables and continuous variables, respectively. I is an appropriate index set, g_i are functions $\{0, 1\}^m \times \mathbb{R}^n \to \mathbb{R}$ being convex on the convex closure of the feasibility domain.

The dual problem to this mixed-integer problem reads:

$$\max_{\lambda\geq 0} \min_{(b,c)} \left\{ g_0(b, c) + \sum_{i\in I} \lambda_i \left(d_i - g_i(b, c)\right) \right\} . \tag{2.78}$$

This problem is convex in d_i and non-differentiable due to the binary variables, because it is the maximum of affine linear functions $m_\lambda + \lambda^T d$ with

$$m_\lambda := \min_{(b,c)} g_0(b, c) - \lambda^T g(b, c) \,. \tag{2.79}$$

Let $B(\lambda)$ define the binary part of the set of optimal points for the inner minimization problem for a given λ:

$$B(\lambda) := \arg \min_{b \in \{0,1\}^m} \left\{ \min_{c \in \mathbb{R}^n} g_0(b,c) + \sum_{i \in I} \lambda_i \left(d_i - g_i(b,c) \right) \right\} . \tag{2.80}$$

The parameter λ has influence on the objective function only, the set of feasible points is not affected. If the functions g_0 and g are linear and $b \in [0,1]^n$, then the solution is stable with respect to a partition of the parameter space, and the partition comprises convex cones. A solution corresponds to the cone the solution is optimal for. This is generalized in [BGK$^+$82] for nonlinear parametric problems. But we are just interested in the continuity with respect to the binary part of the solution.

For fixed b the objective value of the remaining perturbed problem

$$v_b \; := \; \min_{c \in \mathbb{R}^n} g_0(b,c) + \sum_{i \in I} \lambda_i \left(d_i - g_i(b,c) \right) \tag{2.81}$$

is continuous in λ, because the feasibility set is not affected by the parameter (cf.[BGK$^+$82]). The set $B(\lambda)$ contains the elements b, which correspond to minimal v_b. Therefore, the mapping $\lambda \to B(\lambda)$ is *upper semi-continuous according to Berge*. Since $B(\lambda)$ is a subset of $\{0,1\}^m$ the upper semi-continuity reads:

$$\forall \lambda^0 \, \exists \, \delta \; : \; \forall \lambda \in \lambda^0 + \delta I\!B \; : \; B(\lambda) \subseteq B(\lambda^0) \, . \tag{2.82}$$

Hence, no new schedules b have to be taken into account, if the dual parameter varies slightly.

The dual parameter is strictly greater than zero, if the fulfillment of the corresponding constraint needs to be enforced. These non-zero parameters carry information about primal points. For many problems such as scheduling problems, the dependency of $B(\lambda)$ on λ is of more interest than the dependency of the continuous variables on the dual variables.

Problem 2.9 (Binary Problem)

$$\min \{ g_0(b) | \forall i \in I \; : \; g_i(b) \geq d_i \} \tag{2.83}$$

The dual function and the set $B(\lambda)$ are defined as above without c.

Theorem 2.2 (Necessary Optimality Condition) Let $\lambda^\star$ denote the optimal dual parameter, realizing the maximum.
$\Longrightarrow$ Then, for non-zero dual parameters $\lambda_i^\star$ there exist schedules $\tilde{b}$ and $\bar{b}$ enclosing d_i, i.e.:

$$\forall i, \lambda_i^\star > 0 \Longrightarrow \exists \, \tilde{b}, \bar{b} \in B(\lambda^\star) \; : \; g_i(\tilde{b}) \leq d_i \leq g_i(\bar{b}) \tag{2.84}$$

Proof: The proof is done indirectly. First, it is assumed that for a certain i such a $\tilde{b}$ does not exists:

$$\exists i_0,\, \lambda^\star_{i_0} > 0,\, \forall b \in B(\lambda^\star)\, :\, g_{i_0}(b) > d_{i_0}. \tag{2.85}$$

That means $\lambda^\star_{i_0}(d_{i_0} - g_{i_0}(b))$ is strictly negative for all b in $B(\lambda^\star)$. Due to the upper semi-continuity of $B(.)$ this is even valid for a slightly changed λ, i.e. for $\tilde{\lambda} := \lambda^\star - \epsilon e_{i_0}$, when ϵ is small enough, and e_i is the i-th unit vector. Hence:

$$\forall \epsilon > 0,\, B(\lambda^\star - \epsilon e_{i_0}) \subseteq B(\lambda^\star) \Longrightarrow \forall b \in B(\lambda^\star - \epsilon e_{i_0})\, :\, (d_{i_0} - g_{i_0}(b)) < 0\,. \tag{2.86}$$

Then, the value $\tilde{v} := -\max_{b \in B(\lambda^\star)} \epsilon(d_{i_0} - g_{i_0}(b))$ denotes the increment of the value of the dual problem, if $\lambda^\star$ is replaced by $\tilde{\lambda}$. This contradicts the optimality of $\lambda^\star$. Therefore, the assumption $\forall b \in B(\lambda^\star)\, :\, g_{i_0}(b) > d_{i_0}$ is not true.

The proof of the existence of $\bar{b}$ is done similarly. #

Hence, a set of primal points corresponds to an optimal solution of the dual problem, and this set is neither empty nor a singleton. Moreover, after solving the dual problem one has to determine the set of dual preferred primal points. This set might contain a point that is nearly optimal.

Since the dual (Lagrange) information is used to reduce the set of points examined in order to find a nearly optimal point, this method is called Lagrangian reduction.

2.6 Facet Search

> The solution of the dual problem to a certain primal problem gives optimal Lagrange parameters, but no feasible points. In most situations the user is interested in optimal primal points too. The primal points that are the solution of the last inner minimization problem, form a good initial guess for an optimal point. Moreover, if this primal point is feasible with respect to all constraints, then this point is optimal (cf. Theorem 2.1). However, this happens very seldom. Then, some heuristics[4] are performed resulting in a point that is sufficiently close to the optimum. Such heuristics have a long tradition in power optimization, see [ZG88]. In this paper a new heuristics is developed, that uses the dependency between the dual parameters and the set of preferred primal binary points.

Since heuristics are based on background information about the underlying "real-world" problem, the problem considered in this section provides more structure than the problems investigated before.

[4]The heuristics presented in this section was developed for the hydro-thermal power generation system of Chapter 4.

Problem 2.10 (Stochastic Generation Problem) Let T denote the number of time intervals and I the number of thermal units, respectively. Then, the objective function reads:

$$\min_{\boldsymbol{u},\boldsymbol{p},\boldsymbol{x}} \mathbb{E} \sum_{t=1}^{T} \sum_{i=1}^{I} F_i(\boldsymbol{u}_i^t, \boldsymbol{p}_i^t) + S_i^t(\boldsymbol{u}_i)\,, \tag{2.87}$$

where F_i denotes the fuel cost function of the unit i and satisfies the following conditions:

$$u = 0 \implies F_i(u,p) = 0 \tag{2.88}$$

$$u = 1 \implies F_i(u,p) > 0 \tag{2.89}$$

$$u = 1 \wedge \hat{p} > \tilde{p} \implies F_i(u,\hat{p}) > F_i(u,\tilde{p})\,. \tag{2.90}$$

The startup costs $S_i^t(u_i)$ may depend on the duration of preceeding up and down times. The only restriction is:

$$\forall u \,:\, S_i^t(u) \geq 0\,.$$

The constraints for this problem are:

$$\forall i = 1,\ldots,I,\, t = 1,\ldots,T: \quad \boldsymbol{u}_i^t p_i^{min} \leq \boldsymbol{p}_i^t \leq \boldsymbol{u}_i^t p_i^{max} \tag{2.91}$$

$$\boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b} \tag{2.92}$$

$$\forall t = 1,\ldots,T: \quad \sum_{i=1}^{I} \boldsymbol{p}_i^t + \sum_{j\in J} \boldsymbol{x}_j^t \geq \boldsymbol{d}^t \tag{2.93}$$

$$\forall t = 1,\ldots,T: \quad \sum_{i=1}^{I} \boldsymbol{u}_i^t p_i^{max} - \boldsymbol{p}_i^t \geq \boldsymbol{r}^t\,, \tag{2.94}$$

where $\boldsymbol{x}$ denotes some opportunity to shift demand to other time periods, by means of (2.93). Examples for this are pumped hydro storage plants or contracts. The inequalities (2.92) express constraints on such actions.

Let $\boldsymbol{\lambda}^\star$ and $\boldsymbol{\mu}^\star$ denote the optimal Lagrange parameters for the dual problem to problem 2.10:

$$\max_{\boldsymbol{\lambda},\boldsymbol{\mu}\geq 0} \min_{\boldsymbol{u},\boldsymbol{p},\boldsymbol{x}} \mathbb{E} \sum_{t=1}^{T} \quad \boldsymbol{\lambda}^t \left\{ \boldsymbol{d}^t - \sum_{i=1}^{I} \boldsymbol{p}_i^t + \sum_{j\in J} \boldsymbol{x}_j^t \right\} \tag{2.95}$$

$$+ \quad \boldsymbol{\mu}^t \left\{ \boldsymbol{r}^t - \sum_{i=1}^{I} (\boldsymbol{u}_i^t p_i^{max} - \boldsymbol{p}_i^t) \right\} \tag{2.96}$$

$$+ \quad \sum_{i=1}^{I} F_i(\boldsymbol{u}_i^t, \boldsymbol{p}_i^t) + S_i^t(\boldsymbol{u}_i) \tag{2.97}$$

subject to:

$$\forall i = 1,\ldots,I,\, t = 1,\ldots,T: \quad \boldsymbol{u}_i^t p_i^{min} \leq \boldsymbol{p}_i^t \leq \boldsymbol{u}_i^t p_i^{max} \tag{2.98}$$

$$\boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b} \tag{2.99}$$

Usually, the values $\boldsymbol{d}$ and $\boldsymbol{r}$ do not coincide with the output generated by the solution of the inner problem (2.95) – (2.99). That means, a duality gap exists or the solution of the inner problem is not feasible. The infeasibility of primal points obtained during the last iteration forces a search among other primal points. The dual information can be used to reduce this search.

Similarly to $B(\lambda)$ in the Section 2.5 the set $U(\boldsymbol{\lambda}, \boldsymbol{\mu})$ is defined by:

$$U(\boldsymbol{\lambda}, \boldsymbol{\mu}) := \arg\min_{\boldsymbol{u}} \left\{ \min_{\boldsymbol{p}_i} \mathbb{E} \sum_{t=1}^{T} \quad FC_i(\boldsymbol{p}_i^t, \boldsymbol{u}_i^t) + SC_i^t(\boldsymbol{u}_i) \right. \tag{2.100}$$
$$\left. -(\boldsymbol{\lambda}^t - \boldsymbol{\mu}^t)\, \boldsymbol{p}_i^t - \boldsymbol{\mu}^t \boldsymbol{u}_i^t p_i^{max} \right\}.$$

This set can be understood as a set of preferred schedules of independent power producers. These power producers are price takers. They sell power ($\boldsymbol{p}_i^t$ or $\boldsymbol{x}_j^t$) and reserve capacity ($\boldsymbol{u}_i^t p_i^{max} - \boldsymbol{p}_i^t$) to the market for the prices $\boldsymbol{\lambda}^t$ and $\boldsymbol{\mu}^t$, respectively. In order to maximize their gains they prefer certain schedules. Owners of thermal units that use cheap fuel, will produce power, because the market price is usually higher than their costs. Units with high operation costs are in operation only, if their reserve capacities are requested. Otherwise they are offline. Only few[5] power producers are in a situation, where the market prices do not lead to a unique schedule. A "magic hand" should guide these producers in a way, that the system covers demand and reserve.

The heuristics bases on a sequence of schedules within the set $U(\boldsymbol{\lambda}, \boldsymbol{\mu})$. There exists a canonical partial order[6] on such sets, i.e.:

$$\hat{u} \preceq \tilde{u} \Longleftrightarrow \forall i = 1, \ldots, I,\ \forall t = 1, \ldots, T\ :\ \hat{u}_i^t \leq \tilde{u}_i^t.$$

The case $\hat{u} \preceq \tilde{u}$ with $\exists i,\ \exists t\ :\ \hat{u}_i^t < \tilde{u}_i^t$ is denoted by $\hat{u} \prec \tilde{u}$.

The sequence starts with a maximal element $\boldsymbol{u}^{(0)}$ with respect to the partial order $\preceq$. In each iteration a tuple (i, t, ω) is chosen such, that $u_i^t(\omega)$ is still undecided. The corresponding element of the sequence is the maximal element $\boldsymbol{u}^{(l)} \in U(\boldsymbol{\lambda}, \boldsymbol{\mu})$ with $\boldsymbol{u}^{(l)} \preceq \boldsymbol{u}^{(l-1)}$ and ${\boldsymbol{u}^{(l)}}_i^t(\omega) = 0$.

The sequence decreases component-wise, until all undecided pairs (unit, time period) are offline. However, it is possible to abort the sequence evaluation, since the tail of the sequence comprises infeasible problems only.

[5] The power producers in this situation are supposed to have identical configurations, because different configurations lead to different costs and then the market is capable to distinguish these units and the prices are settled accordingly.

[6] The definition is given without the stochasticity of the variables.

Corollary 2.1 Let $\{\boldsymbol{u}^{(l)}\}_{l=0}^{L}$ denote the sequence as constructed above. The set $F(\boldsymbol{u})$ defines the feasibility set:

$$F(\boldsymbol{u}) := \{(\boldsymbol{p}, \boldsymbol{x}) | (\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{x}) \text{ is feasible w.r. to (2.91) – (2.94)}\}$$

Then, the following implication holds:

$$\forall k, F(\boldsymbol{u}^{(k)}) = \emptyset \Longrightarrow \forall l > k \ : \ F(\boldsymbol{u}^{(l)}) = \emptyset$$

Proof: Assuming that there exists $k, l, k < l$ with $F(\boldsymbol{u}^{(k)}) = \emptyset$ and $F(\boldsymbol{u}^{(l)}) \neq \emptyset$. For simplicity, $\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{p}}, \tilde{\boldsymbol{x}}$ are the variables of iteration l, and $\hat{\boldsymbol{u}}, \hat{\boldsymbol{p}}, \hat{\boldsymbol{x}}$ correspond to iteration k. Since $F(\tilde{\boldsymbol{u}})$ was not empty, a point $(\tilde{\boldsymbol{p}}, \tilde{\boldsymbol{x}}) \in F(\tilde{\boldsymbol{u}})$ exists.

Now, $\hat{\boldsymbol{p}}$ is defined by:

$$\hat{p}_i^t(\omega) = \begin{cases} \tilde{u}_i^t p_i^{min} & \text{iff } \tilde{u}_i^t(\omega) < \hat{u}_i^t(\omega) \\ \tilde{p}_i^t(\omega) & \text{otherwise} \end{cases} \quad . \tag{2.101}$$

Taking $\hat{\boldsymbol{x}} = \tilde{\boldsymbol{x}}$, the point $(\hat{\boldsymbol{u}}, \hat{\boldsymbol{p}}, \hat{\boldsymbol{x}})$ is feasible, because of the feasibility of $(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{p}}, \tilde{\boldsymbol{x}})$ and

$$\forall t = 0, \ldots, T : \quad \sum_{i=0}^{I} \hat{\boldsymbol{p}}_i^t \geq \sum_{i=0}^{I} \tilde{\boldsymbol{p}}_i^t \tag{2.102}$$

$$\sum_{i=0}^{I} \hat{\boldsymbol{u}}_i^t p_i^{max} - \hat{\boldsymbol{p}}_i^t \geq \sum_{i=0}^{I} \tilde{\boldsymbol{u}}_i^t p_i^{max} - \tilde{\boldsymbol{p}}_i^t \quad . \tag{2.103}$$

Therefore $(\hat{\boldsymbol{p}}, \hat{\boldsymbol{x}}) \in F(\boldsymbol{u}^{(k)})$ holds, which contradicts the assumption. #

> The union $\cup_{\boldsymbol{u} \in U(\boldsymbol{\lambda}, \boldsymbol{\mu})} F(\boldsymbol{u})$ forms a regular figure in a certain high dimensional space. Usually, this figure has a nonempty interior. The facets as extremal points of this figure are interesting, since they could contain points with a reasonably good objective value. The slopes of these facets differ only slightly. That might be the reason for the behavior of the dual algorithm. At the beginning the dual value increases much. After a certain number of iterations the increase is no longer significant, but the dual point still moves. The geometric interpretation means, that many facets are active or almost active during the final iterations. This could be imagined like the surface of a diamond near the point where all colors of the rainbow are to be seen.

The heuristics "facet search" does not consider all facets, only facets of a certain "path" are investigated. This path goes almost diagonally through these facets, and will hopefully hit the best. Then, the algorithm of this heuristics reads:

1. Solve the dual problem and get optimal Lagrange parameters $(\boldsymbol{\lambda}, \boldsymbol{\mu})$.
2. Detect $U(\boldsymbol{\lambda}, \boldsymbol{\mu})$ by (2.100).

3. Set first schedule $\boldsymbol{u}^{(0)} \in \arg\max_{\preceq} U(\boldsymbol{\lambda}, \boldsymbol{\mu})$. Usually, $\boldsymbol{u}^{(0)}$ has more units in operation than are needed in order to cover the demand. Moreover, the corresponding economic dispatch problem contains feasible points.

4. Set $v^\star = \infty$, $k = 0$.

5. Solve the economic dispatch problem with $\boldsymbol{u} = \boldsymbol{u}^{(k)}$:

$$v^{(k)} = \min_{(\boldsymbol{p},\boldsymbol{x})\in F(\boldsymbol{u})} \mathbb{E} \sum_{t=1}^{T} \sum_{i=1}^{I} F_i(\boldsymbol{u}_i^t, \boldsymbol{p}_i^t) + S_i^t(\boldsymbol{u}_i)\,. \tag{2.104}$$

   (a) If $F(\boldsymbol{u}^{(k)}) = \emptyset$, goto 8, because the search for primal points can be aborted due to corollary 2.1.

   (b) If $v^{(k)} < v^\star$, then $v^\star := v^{(k)}$ and $\boldsymbol{u}^\star := \boldsymbol{u}^{(k)}$.

6. Detect a time period t, a scenario ω, and an undecided unit i. The next schedule is determined by

$$\boldsymbol{u}^{(k+1)} \in \arg\max_{\preceq} \left\{ \boldsymbol{u} \in U(\boldsymbol{\lambda}, \boldsymbol{\mu}) | \boldsymbol{u} \preceq \boldsymbol{u}^{(k)} \wedge u_i^t(\omega) = 0 \right\}$$

7. Goto 5

8. The best element of the sequence, i.e. $v^\star = \arg\max_k v^{(k)}$, provides the primal point $(\boldsymbol{u}^\star, \boldsymbol{p}^\star, \boldsymbol{x}^\star)$ with the best objective value as the the result of this heuristics. **STOP**.

In step 5 an economic dispatch problem has to be solved. Since it has to be solved for each element of the sequence, an efficient method is required. The subsequent dispatch problems differ in few binary variables only. Therefore, iteration methods are superior.

In point 6 some knowledge can be used in order to obtain a good result. The method used to determine the node and the unit that is going to be switched off, has a great influence on the result.

Chapter 3

Algorithms for Stochastic Subproblems

The stochastic Lagrangian relaxation method decomposes the original problem into stochastic subproblems. They have to be solved several times, since they are part of the inner loop of the solver for the dual problem. Therefore, the stochastic Lagrangian relaxation method needs efficient solvers for these problems in order to solve the problem in a reasonable amount of time.

3.1 Descent Method for Stochastic Storage Problems

Many multi-stage stochastic linear programs have a simple structure. For example, the stages are coupled by some resource state variables. This dynamic structure can be exploited. Similar to dynamic programming the expected cost-to-go is just a function of the current state. Instead of evaluating the state space the presented descent method looks for descent directions for the expected cost-to-go. This section shows that it is sufficient to examine just a small but sufficiently large subset of directions.

3.1.1 The Model

The model includes just one exogenous stochastic variable:

$$\lambda : (\Omega, \mathcal{A}, \mathbb{P}) \times \{1, \ldots, T\} \longrightarrow \mathbb{R}\,.$$

The decision variable x and the resource state variable l are stochastic too:

$$x, l : \Omega \times \{1, \ldots, T\} \longrightarrow \mathbb{R}\,.$$

The bold notation (i.e. $\boldsymbol{x}^t$) is used to denote the random variable $x(.,t)$ for the aim of simplicity.

Problem 3.1 Then, the stochastic storage problem reads:

$$\min_{\boldsymbol{x}} \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \boldsymbol{x}^t \tag{3.1}$$

subject to:

$$\forall t = 1 \dots T: \quad x^{min} \leq \boldsymbol{x}^t \leq x^{max} \tag{3.2}$$

$$0 \leq \boldsymbol{l}^t \leq l^{max} \tag{3.3}$$

$$\boldsymbol{l}^t = \boldsymbol{l}^{t-1} + \boldsymbol{x}^t \tag{3.4}$$

$$\boldsymbol{l}^0 = l^{in}, \quad \boldsymbol{l}^T = l^{end} \tag{3.5}$$

And the decision variable $\boldsymbol{x}$ and the resource state variable $\boldsymbol{l}$ have to be nonanticipative, i.e. $\mathcal{F}(\boldsymbol{x}, \boldsymbol{l}) \subseteq \mathcal{F}(\boldsymbol{\lambda})$.

Proposition 3.1 (Existence of an optimal point) Since the constraints have a simple structure, necessary conditions on the existence of a feasible point are quite simple:

- $0 \leq l^{in} \leq l^{max}$
- $0 \leq l^{end} \leq l^{max}$
- $Tx^{min} \leq l^{end} - l^{in} \leq Tx^{max}$.

$\Longrightarrow$ Then, there exists a feasible point.

It is also possible to formulate the problem without the variable $\boldsymbol{l}$ and express the restrictions (3.3), (3.4), and (3.5) just using the $\boldsymbol{x}$-, l^{in}-, l^{end}-, and l^{max}- variables. The introduction of $\boldsymbol{l}$ simplifies the constraints — they get a Markovian structure. This plays an important part in the following considerations.

> Such a model appears for instance as a subproblem in the short term optimization of a hydro-thermal power generation system. In this case $\boldsymbol{l}$ denotes the stored water at the upper dam, while $\boldsymbol{x}$ is the corresponding release of water or the pumped amount, if $\boldsymbol{x}$ is positive or negative, respectively. Then, $\boldsymbol{\lambda}^t$ denotes the value of releasing or pumping water. One can also think of it as a model for trading goods, stocks etc. Then, $\boldsymbol{l}$ denotes the portfolio and $\boldsymbol{x}$ the buy/sale decisions.

In this problem the storage volume variable $\boldsymbol{l}_j^t$ plays the part of a resource state variable, that means the variables for $t > t_0$ and $t < t_0$ do not influence each other when $\boldsymbol{l}_j^t$ for $t = t_0$ is fixed. The equation (3.4) describing the dynamics of the system is one-dimensional. Hence, the storage volume can be increased and decreased using $\boldsymbol{x}$. The costs of changing the storage volume at time $t = t_0$, i.e., $\boldsymbol{\lambda}^{t_0}\Delta\boldsymbol{x}^{t_0}$, have to be compared with the changes of costs in all subsequent time periods,

$$\mathbb{E}\sum_{t=t_0+1}^{T}\boldsymbol{\lambda}^t\Delta\boldsymbol{x}^t, \tag{3.6}$$

in order to find out whether an alteration of the storage volume leads to a decrease of the objective function or not. If such a change of the storage volume $\boldsymbol{l}_j^t$ in any node does not lead to a decrease of the objective function, then the current point $(\boldsymbol{x}, \boldsymbol{l})$ is optimal for (3.1)-(3.5). The subsequent costs (3.6) are caused by changes of variables in the g-subtree in order to satisfy the balance (3.4). Since the problem has a special structure, certain points $\Delta\boldsymbol{x}^t$ yielding a minimal value of (3.6) have many zero components. In [Now96] it is shown that the search for descent directions may be restricted to such elements. Moreover, the non-zero components describe a g-subtree of the scenario tree. Then, the conditions on step lengths and on steps to be descent directions take simpler forms. The construction of these g-subtrees is done in a systematic way starting at the leaves and determining which nodes should be leaves in such g-subtrees. This is explained on an example next.

3.1.2 Example for a Simple Direction

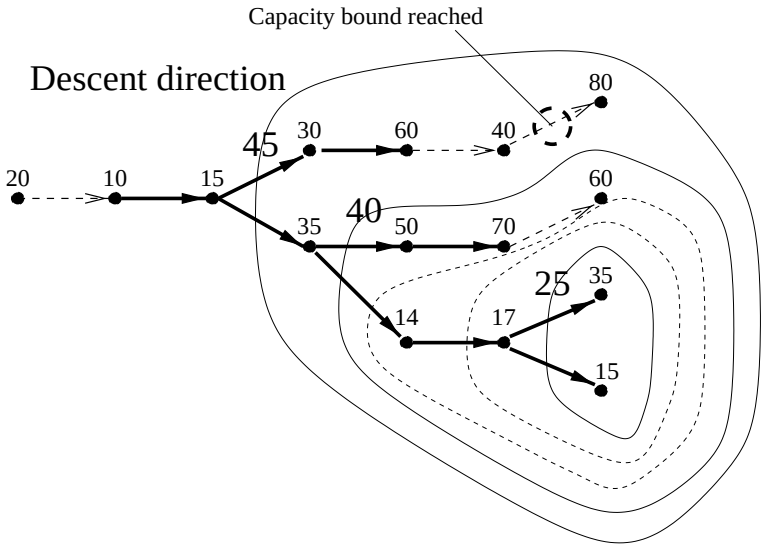

Figure 3.1: Example of a simple direction

Figure 3.1 shows an example with 4 scenarios having identical probabilities (i.e. 1/4) and 7 stages. The small numbers at the nodes represent the values $\boldsymbol{\lambda}^t$. The g-subtree mentioned above is marked with thick arrows (starting at the node with value 10 and ending at nodes with 60, 70, 35, and 15 as values).

Problem 3.2 To simplify the presentation, we take up the position of the storage operator. Then, the $\boldsymbol{\lambda}$-values represent prices for buying and selling a certain commodity, and the aim is to maximize the profit, i.e.:

$$\max_{\boldsymbol{x}} \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \boldsymbol{x}^t \tag{3.7}$$

subject to:

$$\forall t = 1 \ldots T: \qquad x^{min} \leq \boldsymbol{x}^t \leq x^{max},\ 0 \leq \boldsymbol{l}^t \leq l^{max} \tag{3.8}$$

$$\boldsymbol{l}^t = \boldsymbol{l}^{t-1} + \boldsymbol{x}^t \tag{3.9}$$

$$\boldsymbol{l}^0 = l^{in}, \quad \boldsymbol{l}^T = l^{end}. \tag{3.10}$$

Assume that a certain amount is bought at the second stage to a price of 10. The price paid at this node has to be compared with the gain from selling the amount at some nodes in the g-subtree in order to keep the balance (3.4). Each node in the g-subtree is examined in order to determine whether the amount should be sold at this node or should be kept for the subsequent nodes.

If the amount is kept up to the last stage, it has to be sold in any case. In our example, the gain is 15 in the lowest scenario. If this happens, the amount is also sold at the last stage of the second lowest scenario due to the stochastic nature of the problem. Hence, the gain is 35. The average gain of these two scenarios is 25, shown at the lowest inner solid surrounding. The comparison of this average gain with the price at the node before, i.e. 17, leads to the decision to keep the amount up to the last stage. Hence, it follows that the gain of selling the amount at this node or later is 25. This is denoted by a surrounding with a dotted line, indicating that it has the same value as the inner one. The decision at the node before, i.e., the result of the comparing the value of the surrounded g-subtree (25) with the value at the node (14), leads to the same decision (to keep the amount) save for the fact that, at this point, it is out of interest at which node the amount is actually sold. In the last but one stage of the second scenario the comparison of the value for the last stage (60) with the one of the stage before (70) yields the decision to sell the amount at that node, i.e. at the last but one stage.

The uppermost scenario indicates the case where keeping up to the last stage is not feasible due to capacity bounds. Hence, the comparisons of the nodes before indicate that the amount should be sold at the node with value 60. Applying the same analysis to all nodes yields where the amount should be kept and where it should be sold in order to get maximal gain from buying at the second stage. A

flow from the second stage to subsequent stages is associated with this maximal gain, which corresponds to the thick marked g-subtree in Figure 3.1. Note that the leaves of this g-subtree correspond to nodes where the decision is selling. Further, these decisions are independent of the node at which this g-subtree starts. In case the storage is not empty at the first stage, it is also feasible to sell first and to buy back the amount later. However, this can be treated in a similar way and leads to a second set of binary decisions. After this analysis has been applied to all nodes of the scenario tree, an ascent direction for the maximization problem can be found examining all nodes just once.

This analysis can be carried out for the minimization problem (3.1)-(3.5) in a similar way.

3.1.3 Conditions on Directions to be Descent Directions

In this section directions are investigated to determine whether they are descent directions or not. Since the resource state variable $\boldsymbol{l}$ is well determined by the initial value l^{in} and the variable $\boldsymbol{x}$, changes of the $\boldsymbol{l}$ variable are not considered explicitly.

For the minimization problem (3.1)-(3.5) the conditions on the existence of a descent direction are sketched next. The variables and decisions for the case of an increased storage are denoted by the superscript *up*, while *down* refers to the case of a decreased storage. The decision to reduce the storage is denoted by $b_k^{up} = 1$, while $b_k^{up} = 0$ refers to the decision to keep the additional amount. Similarly, the notations $b_k^{down} = 1$ and $b_k^{down} = 0$ are used. Let π_k be the probability and $Succ(k)$ the set of all successors of the node k, and introduce the following auxiliary variables:

- d_k^{up} and d_k^{down} denote upper bounds for the step length:

$$d_k^{up} = \begin{cases} x_k - x^{min}, & \text{if } b_k^{up} = 1 \\ \min\{l^{max} - l_k, \min_{\kappa \in Succ(k)} d_\kappa^{up}\}, & \text{if } b_k^{up} = 0 \end{cases} \tag{3.11}$$

$$d_k^{down} = \begin{cases} x^{max} - x_k, & \text{if } b_k^{down} = 1 \\ \min\{l_k, \min_{\kappa \in Succ(k)} d_\kappa^{down}\}, & \text{if } b_k^{down} = 0 \end{cases} . \tag{3.12}$$

- r_k^{up} and r_k^{down} denote the best average values for the subtrees:

$$r_k^{up} = \begin{cases} \lambda_k \pi_k, & \text{if } b_k^{up} = 1 \\ \sum_{\kappa \in Succ(k)} r_\kappa^{up}, & \text{if } b_k^{up} = 0 \end{cases} \tag{3.13}$$

$$r_k^{down} = \begin{cases} \lambda_k \pi_k, & \text{if } b_k^{down} = 1 \\ \sum_{\kappa \in Succ(k)} r_\kappa^{down}, & \text{if } b_k^{down} = 0 \end{cases} . \tag{3.14}$$

For the leaves of the scenario tree the variables b_k^{up} and b_k^{down} must have the value 1, otherwise these variables are not well defined. The correctness of all nodes corresponding to the time period $t+1$ yields the correct definition of the stage-t variables. Hence, all these auxiliary variables are well defined.

Now, the conditions on a direction to be a descent direction, to which a g-subtree starting at node k is associated, reads:

$$\text{Case of increasing the level:} \qquad \min\{x^{max} - x_k, d_k^{up}\}\{\lambda_k \pi_k - r_k^{up}\} \leq 0\,, \quad (3.15)$$

$$\text{Case of decreasing the level:} \quad \min\{x_k - x^{min}, d_k^{down}\}\{-\lambda_k \pi_k + r_k^{down}\} \leq 0\,. \quad (3.16)$$

Theorem 3.1 Let $(\boldsymbol{x}, \boldsymbol{l})$ and $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ be feasible points with respect to (3.2)-(3.5).
$\Longrightarrow$ Then, there exists a feasible point $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}})$ with the following property:

(P1) The storage level process $\tilde{\boldsymbol{l}}$ of the new point $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}})$ is bounded by the old storage levels, i.e.:

$$\forall \nu \in \Omega,\, \forall t = 1, \ldots, T \,:\, \tilde{l}_{([\nu]_t, t)} \in [\min\{l_{([\nu]_t,t)}, \hat{l}_{([\nu]_t,t)}\}, \max\{l_{([\nu]_t,t)}, \hat{l}_{([\nu]_t,t)}\}].$$

Proof: Let $(\Delta \boldsymbol{x}, \Delta \boldsymbol{l})$ be the difference $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}}) - (\boldsymbol{x}, \boldsymbol{l})$. If this difference is zero, then one can take $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}}) = (\boldsymbol{x}, \boldsymbol{l})$ and the property (P1) is fulfilled trivially. Assume now that this difference is not zero. This means at least one component Δx_k is not zero, thus it follows:

$$\exists\, t_0 \in \arg\min \left\{ t \mid \exists\, \nu \in \Omega,\ \Delta x_{([\nu]_t, t)} \neq 0 \right\}.$$

Let ω be a random element such that $\Delta x_{([\omega]_{t_0}, t_0)} \neq 0$. This node $k = ([\omega]_{t_0}, t_0)$ is the root of the g-subtree which is constructed next. For each element $\nu \in [\omega]_{t_0}$ there exists a time period $t_1(\nu) := \arg\min \left\{ t \mid \Delta x_{([\omega]_{t_0}, t_0)} \Delta x_{([\nu]_{t_1}, t_1)} \leq 0 \right\}$, because the balance equation is fulfilled by $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ and $(\boldsymbol{x}, \boldsymbol{l})$. The leaves of the g-subtree are the nodes $L = \{([\nu]_{t_1(\nu)}, t_1(\nu))\}_{\nu \in [\omega]_{t_0}}$. Then, the step-length is determined by:

$$d = \min_{k \in L \cup \{([\omega]_{t_0}, t_0)\}} |\Delta x_k| \quad (3.17)$$

In case of $\Delta x_{([\omega]_{t_0}, t_0)} > 0$ the new process $\tilde{\boldsymbol{x}}$ is constructed as:

$$\tilde{x}_{([\nu]_t, t)} = \begin{cases} x_{([\nu]_t, t)} + d & \text{iff } \nu \in [\omega]_{t_0} \wedge t = t_0 \\ x_{([\nu]_t, t)} - d & \text{iff } \nu \in [\omega]_{t_0} \wedge t = t_1(\nu) \\ x_{([\nu]_t, t)} & \text{otherwise.} \end{cases}$$

In case of $\Delta x_{([\omega]_{t_0}, t_0)} < 0$ the construction of $\tilde{\boldsymbol{x}}$ is similarly:

$$\tilde{x}_{([\nu]_t, t)} = \begin{cases} x_{([\nu]_t, t)} - d & \text{iff } \nu \in [\omega]_{t_0} \wedge t = t_0 \\ x_{([\nu]_t, t)} + d & \text{iff } \nu \in [\omega]_{t_0} \wedge t = t_1(\nu) \\ x_{([\nu]_t, t)} & \text{otherwise.} \end{cases}$$

The proof of the feasibility of the process $\tilde{\boldsymbol{x}}$ is done scenario-wise, because only scenarios with $\nu \in [\omega]_{t_0}$ are affected, i.e.: for $\nu \notin [\omega]_{t_0}$ the corresponding components of $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ do not differ. The same is valid for the first few components corresponding to time periods until t_0, i.e.:

$$\forall t = 1, \ldots, t_0 - 1 \; : \; x_{([\omega]_t, t)} = \tilde{x}_{([\omega]_t, t)}.$$

Without loss of generality, the case $\Delta x_{([\omega]_{t_0}, t_0)} > 0$ is assumed, the case $\Delta x_{([\omega]_{t_0}, t_0)} < 0$ can be considered similarly. Under this assumption and for the time period t_0 the value of $\tilde{x}_{([\omega]_{t_0}, t_0)}$ is contained in the interval $[x_{([\omega]_{t_0}, t_0)}, \hat{x}_{([\omega]_{t_0}, t_0)}]$. For the intermediate time periods the components of $\tilde{\boldsymbol{x}}$ and of $\boldsymbol{x}$ do not differ, i.e.:

$$\forall \nu \in [\omega]_{t_0}, \forall t = t_0 + 1, \ldots, t_1(\nu) - 1 \; : \; \tilde{x}_{([\nu]_t, t)} = x_{([\nu]_t, t)}.$$

For the time period $t_1(\nu)$ and for the case $\Delta x_{([\omega]_{t_0}, t_0)} > 0$ the value of $\tilde{x}_{([\nu]_{t_1(\nu)}, t_1(\nu))}$ is contained in the interval $[\hat{x}_{([\nu]_{t_1(\nu)}, t_1(\nu))}, x_{([\nu]_{t_1(\nu)}, t_1(\nu))}]$. For the remaining time periods the same is valid as for the intermediate time periods, i.e.:

$$\forall \nu \in [\omega]_{t_0}, \forall t = t_1(\nu) + 1, \ldots, T \; : \; \tilde{x}_{([\nu]_t, t)} = x_{([\nu]_t, t)}.$$

Hence, $\tilde{\boldsymbol{x}}$ is feasible.

To this process $\tilde{\boldsymbol{x}}$ a storage process $\tilde{\boldsymbol{l}}$ is associated due to (3.4) and (3.5). Next, the feasibility of $\tilde{\boldsymbol{l}}$ is to be proved. Again, this is done scenario-wise, because scenarios $\nu \notin [\omega]_{t_0}$ are out of interest. For the first few time periods, the components of $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ do not differ, this results in:

$$\forall t = 1, \ldots, t_0 - 1 \; : \; l_{([\omega]_t, t)} = \tilde{l}_{([\omega]_t, t)}.$$

The next time periods have a slightly more complex structure, here the relation to both processes $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ have to be taken into account.

Without loss of generality the case $\Delta x_{([\omega]_{t_0}, t_0)} > 0$ is considered, since the other case can be treated similarly. In that case the storage level $\hat{l}_{([\omega]_{t_0}, t_0)}$ is greater than $l_{([\omega]_{t_0}, t_0)}$. This gap increases during the next time periods until time $t_1(\nu)$, because this is the first time period in each scenario, where $\Delta x_{([\nu]_{t_1(\nu)}, t_1(\nu))}$ is negative. In comparison to $\boldsymbol{l}$ the level $\tilde{\boldsymbol{l}}$ is shifted by d for the intermediate time periods. Due to equation (3.17) the amount d is small enough such that $\tilde{l}_{([\nu]_t, t)}$ is still less than $\hat{l}_{([\nu]_t, t)}$ for these time periods. Hence it follows:

$$\forall \nu \in [\omega]_{t_0}, \forall t = t_0, \ldots, t_1(\nu) - 1 \; : \; \tilde{l}_{([\nu]_t, t)} \in [l_{([\nu]_t, t)}, \hat{l}_{([\nu]_t, t)}].$$

For the remaining time periods the levels $\tilde{l}_{([\nu]_t, t)}$ and $l_{([\nu]_t, t)}$ coincide. Since $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ and $(\boldsymbol{x}, \boldsymbol{l})$ were feasible with respect to (3.2)-(3.5), the point $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}})$ is feasible too. Furthermore, the storage process has the property (P1). #

Corollary 3.1 (Existence of a converging sequence) Under the assumptions of theorem 3.1 there exists a sequence of points $\{(\tilde{\boldsymbol{x}}^n, \tilde{\boldsymbol{l}}^n)\}$ such that:

1. to the difference between 2 consecutive points a step corresponding to a g-subtree can be assigned
2. the sequence converges to $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ in a finite number of steps.

Proof: The sequence is the result of the repeated application of the procedure described in the proof of theorem 3.1, where the point $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ is fixed and starting with $(\tilde{\boldsymbol{x}}^1, \tilde{\boldsymbol{l}}^1) = (\boldsymbol{x}, \boldsymbol{l})$ the point $(\boldsymbol{x}, \boldsymbol{l})$ is replaced by the current iterate$(\tilde{\boldsymbol{x}}^n, \tilde{\boldsymbol{l}}^n)$. This procedure guarantees point 1.

Next, the verification of point 2 is done. The focus is set on the difference $(\Delta\boldsymbol{x}^n, \Delta\boldsymbol{l}^n) = (\tilde{\boldsymbol{x}}^n, \tilde{\boldsymbol{l}}^n) - (\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$. The absolute value of Δx_k^n decreases, because the new point $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}})$ is constructed in such a way. Equation (3.17) ensures that at least one component of $\Delta\boldsymbol{x}$ vanishes. Hence, in each iteration one component of $\Delta\boldsymbol{x}^n$ attains zero. Because $\Delta\boldsymbol{x}^0$ had a finite number of non-zero components the sequence converges in finitely many steps. #

Corollary 3.2 Let $(\boldsymbol{x}, \boldsymbol{l})$ and $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ be feasible points with respect to (3.2)-(3.5). The sequence of steps $\{\Delta\boldsymbol{x}^n\}_{n=1}^N$ is constructed as in the proof of theorem 3.1. The new point $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}})$ is built using this sequence of steps without the first step, thus $\tilde{\boldsymbol{x}} = \boldsymbol{x} + \sum_{n=2}^N \Delta\boldsymbol{x}^n$ and $\tilde{\boldsymbol{l}} = \boldsymbol{l} + \sum_{n=2}^N \Delta\boldsymbol{l}^n$.
$\Longrightarrow$ Then, the point $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}})$ is feasible.
Proof: Since corollary 3.1 gives

$$\hat{\boldsymbol{x}} = \boldsymbol{x} + \sum_{n=1}^N \Delta\boldsymbol{x}^n \;, \quad \hat{\boldsymbol{l}} = \boldsymbol{l} + \sum_{n=1}^N \Delta\boldsymbol{l}^n \;,$$

the point $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}})$ fulfills:

$$\tilde{\boldsymbol{x}} = \hat{\boldsymbol{x}} - \Delta\boldsymbol{x}^1 \;, \quad \tilde{\boldsymbol{l}} = \hat{\boldsymbol{l}} - \Delta\boldsymbol{l}^1 \;.$$

It is the same point one gets if $(\boldsymbol{x}, \boldsymbol{l})$ and $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ are exchanged in theorem 3.1. Hence, the sequence converges to a feasible points, even if the first step is omitted. #

As a additional result of this corollary, the first consecutive steps can be omitted and still the sequence converges to feasible points.

Corollary 3.3 Let $(\boldsymbol{x}, \boldsymbol{l})$ and $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ be feasible points with respect to (3.2)-(3.5), and $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ is optimal, but $(\boldsymbol{x}, \boldsymbol{l})$ is not.
$\Longrightarrow$ Then, there exists a sequence of steps as constructed in the proof of theorem 3.1, and all steps are descent steps.
Proof: The first step must be a descent step, otherwise this step can be omitted and still the sequence converges to a certain point. Due to corollary 3.2 this point is feasible and it has a lower objective value than $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ if the first step had not been a descent step. This is a contradiction to the optimality of $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$. #

Furthermore, the first consecutive steps can be omitted and then it follows that all steps must be descent steps. As a consequence of this corollary, there exists a sequence of proper descent steps converging to an optimal point, since all non-proper descent steps can be omitted.

Theorem 3.2 Let $(\boldsymbol{x}, \boldsymbol{l})$ and $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ be feasible points with respect to (3.2)-(3.5), and $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{l}})$ is optimal but $(\boldsymbol{x}, \boldsymbol{l})$ is not.
$\Longrightarrow$ Then, there exist values for $\{(b_k^{up}, b_k^{down})\}_{k \in V}$, such that condition (3.15) or condition (3.16) is satisfied.
Proof: As a result of corollary 3.3, there exists a sequence of steps and all steps are descent steps. To prove this theorem the first step is considered. The node $([\omega]_{t_0}, t_0)$ was used as a root node, while the leaves of the considered g-subtree were the nodes $L = \{([\nu]_{t_1(\nu)}, t_1(\nu)) | \nu \in [\omega]_{t_0}\}$. The first step was a descent step, i.e.:

$$\mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \tilde{\boldsymbol{x}}^t - \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \boldsymbol{x}^t = \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \left(\tilde{\boldsymbol{x}}^t - \boldsymbol{x}^t \right) \leq 0, \tag{3.18}$$

where $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{l}})$ denotes the point obtained by the procedure in the proof of theorem 3.1. The difference reads

$$\begin{aligned}
&= \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \left\{ \tilde{\boldsymbol{x}}^t - \boldsymbol{x}^t \right\} \\
&= \left\{ \tilde{x}_{([\omega]_{t_0}, t_0)} - x_{([\omega]_{t_0}, t_0)} \right\} \lambda_{([\omega]_{t_0}, t_0)} \pi_{([\omega]_{t_0}, t_0)} + \sum_{k \in L} \left\{ \tilde{x}_k - x_k \right\} \lambda_k \pi_k \\
&= d \left\{ \lambda_{([\omega]_{t_0}, t_0)} \pi_{([\omega]_{t_0}, t_0)} - \sum_{k \in L} \lambda_k \pi_k \right\},
\end{aligned}$$

if d denotes $\tilde{x}_{([\omega]_{t_0}, t_0)} - x_{([\omega]_{t_0}, t_0)}$, because many components of $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ coincide, and the absolute values of $\tilde{x}_k - x_k$ do not differ for $k = ([\omega]_{t_0}, t_0)$ or $k \in L$. The sum can be rewritten using the auxiliary variables if the values $\{(b_k^{up}, b_k^{down})\}_{k \in V}$ are chosen as follows:

$$b^{up}_{([\nu]_t, t)} = b^{down}_{([\nu]_t, t)} = \begin{cases} 1 & \text{iff } \nu \in [\omega]_{t_0} \wedge t = t_1(\nu) \vee t = T \\ 0 & \text{otherwise.} \end{cases}$$

Then, the following is valid:

$$\sum_{k \in L} \lambda_k \pi_k = r^{up}_{([\omega]_{t_0}, t_0)} = r^{down}_{([\omega]_{t_0}, t_0)}.$$

In case the storage level is increased at the root node of the g-subtree, d is positive and a lower bound for $\min\{x^{max} - x_k, d_k^{up}\}$. This and (3.18) leads to the fulfillment of condition (3.15).

In the other case, i.e. the storage level is decreased at the root node, d is negative and $-d$ is a lower bound for $\min\{x_k - x^{min}, d_k^{down}\}$. Then (3.18) leads to

$$|d| \left\{ -\lambda_{([\omega]_{t_0},t_0)} \pi_{([\omega]_{t_0},t_0)} + \sum_{k \in L} \lambda_k \pi_k \right\} \leq 0,$$

which means, that condition (3.16) is fulfilled. #

Once the condition (3.15) or (3.16) is fulfilled, a descent direction exists. If the inequalities (3.15) and (3.16) are strongly fulfilled , the descent direction is a proper descent direction.

3.1.4 Extensions

Despite the fact that this section considers a simple model, the presented method has a larger capability. The main idea was to construct descent direction in such a way, that the existence could efficiently be proved, and that the steps require just few elementary computations.

Next, few extensions of the basic model are given.

The problem 3.1 does not consider the spread between the purchase price and the selling price like transaction costs in stock markets. Such a difference occurs even in simple storage problems, where more energy has to be used to pump water uphill than regained by turbining due to the restricted efficiency of the machines.

Problem 3.3 The stochastic storage problem with "transaction" costs reads:

$$\min_{\boldsymbol{x}} \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \left(\frac{1}{\eta} (\boldsymbol{x}^t)^+ + (\boldsymbol{x}^t)^- \right) \tag{3.19}$$

subject to:

$$\forall t = 1 \ldots T: \quad x^{min} \leq \boldsymbol{x}^t \leq x^{max} \tag{3.20}$$

$$0 \leq \boldsymbol{l}^t \leq l^{max} \tag{3.21}$$

$$\boldsymbol{l}^t = \boldsymbol{l}^{t-1} + \boldsymbol{x}^t \tag{3.22}$$

$$\boldsymbol{l}^0 = l^{in}, \quad \boldsymbol{l}^T = l^{end}, \tag{3.23}$$

where a^+ and a^- are defined as $a^+ = \max\{a, 0\}$ and $a^- = \min\{a, 0\}$ and $\eta \in (0, 1)$ denotes the limited cycle efficiency of that storage plant.

The objective function of this problem is a piecewise linear function. This nonlinearity can be avoided by splitting the x variable. Then, $\boldsymbol{s} \geq 0$ describes the energy from turbining/selling, while $\boldsymbol{w} \geq 0$ denotes the amount of energy needed

for pumping/buying. Splitting this variable the problem gets a linear structure, i.e.:

Problem 3.4 (Stochastic Pumped Storage Problem)

$$\min_{(\boldsymbol{s},\boldsymbol{w})} \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \left(\boldsymbol{w}^t - \boldsymbol{s}^t \right) \tag{3.24}$$

subject to:

$$\begin{aligned}
\forall t = 1 \ldots T: \quad 0 \leq \quad \boldsymbol{w}^t \quad &\leq w^{max} && (3.25)\\
0 \leq \quad \boldsymbol{s}^t \quad &\leq s^{max} && (3.26)\\
0 \leq \quad \boldsymbol{l}^t \quad &\leq l^{max} && (3.27)\\
\boldsymbol{l}^t \quad &= \quad \boldsymbol{l}^{t-1} + \eta \boldsymbol{w}^t - \boldsymbol{s}^t && (3.28)\\
\boldsymbol{l}^0 = l^{in}, \quad \boldsymbol{l}^T &= l^{end}, && (3.29)
\end{aligned}$$

where $\eta \in (0,1)$ denotes the cycle efficiency of the pumped storage plant.

The introduction of $\boldsymbol{s}$ and $\boldsymbol{w}$ led to a linear structure.

Corollary 3.4 (Complementarity $\boldsymbol{s}/\boldsymbol{w}$) Under the assumption of $\boldsymbol{\lambda} > 0$ a point with $\exists\, t^\star,\ 1 \leq t^\star \leq T,\ \exists \nu \in \Omega\ :\ s^{t^\star}(\nu) w^{t^\star}(\nu) > 0$ is not optimal.

Proof: As discussed in [GRS92] such a point $(\boldsymbol{s}, \boldsymbol{w})$ can be replaced by a point $(\tilde{\boldsymbol{s}}, \tilde{\boldsymbol{w}})$, which has an objective value less than that of $(\boldsymbol{s}, \boldsymbol{w})$.

Assume: $\lambda^\star(\nu) > 0 \wedge s^{t^\star}(\nu) w^{t^\star}(\nu) > 0$, then the point $(\tilde{\boldsymbol{s}}, \tilde{\boldsymbol{w}})$ is defined as:

$$\begin{aligned}
\delta \quad &:= \quad \min\{w^{t^\star}(\nu), \frac{1}{\eta} s^{t^\star}(\nu)\} && (3.30)\\
w^t(\omega) \quad &:= \quad \begin{cases} w^t(\omega) - \delta, & \text{iff } \nu = \omega \wedge t = t^\star \\ w^t(\omega), & \text{otherwise} \end{cases} && (3.31)\\
s^t(\omega) \quad &:= \quad \begin{cases} s^t(\omega) - \eta\delta, & \text{iff } \nu = \omega \wedge t = t^\star \\ s^t(\omega), & \text{otherwise} \end{cases} . && (3.32)
\end{aligned}$$

This construction ensures, that the point $(\tilde{\boldsymbol{s}}, \tilde{\boldsymbol{w}})$ satisfies all bounds. The $\boldsymbol{l}$-variable remains unchanged:

$$\begin{aligned}
l^{t^\star}(\nu) \quad &= \quad l^{t^\star - 1}(\nu) + \eta(w^{t^\star}(\nu) - \delta) - (s^{t^\star}(\nu) - \eta\delta) && (3.33)\\
&= \quad l^{t^\star - 1}(\nu) + \eta w^{t^\star}(\nu) - s^{t^\star}(\nu)\,. && (3.34)
\end{aligned}$$

Then, the difference of the objective values reads:

$$\mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \left(\boldsymbol{w}^t - \boldsymbol{s}^t \right) - \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \left(\tilde{\boldsymbol{w}}^t - \tilde{\boldsymbol{s}}^t \right) = \lambda^{t^\star}(\nu)(1-\eta)\delta P([\nu]_{t^\star}) \tag{3.35}$$

Because the point $(\tilde{\boldsymbol{s}}, \tilde{\boldsymbol{w}})$ is feasible, and the right hand side of equation (3.35) is positive, the points $(\boldsymbol{s}, \boldsymbol{w})$ could not have been optimal.

#

In that case, the problems 3.4 and 3.3 are equivalent by setting

$$\boldsymbol{s}^t = -(\boldsymbol{x}^t)^-,\ \boldsymbol{w}^t = \frac{1}{\eta}(\boldsymbol{x}^t)^+.$$

In order to solve problem 3.4 the different contribution of $\boldsymbol{w}$ and $\boldsymbol{s}$ to the objective function has to be taken into account. In case of $w_k > 0$ for example, λ_k has to be replaced by $\frac{1}{\eta}\lambda_k$ within the definition of r_k^{down} in (3.14). Depending on $w_k > 0$, $w_k \geq 0$, $s_k > 0$, and $s_k \geq 0$ similar changes have to be made in (3.11), (3.12), (3.13), (3.15), and (3.16), respectively. In problem 3.3 the cost function was piecewise linear with just 2 segments, but problems with a more sophisticated objective function can be treated in the same way.

Problem 3.5 (Economic Dispatch Problem) The cost function includes additional variables:

$$\min_{(\boldsymbol{p},\boldsymbol{s},\boldsymbol{w})} \mathbb{E} \sum_{t=1}^{T} \sum_{i=1}^{I} c_i \boldsymbol{p}_i^t \tag{3.36}$$

subject to:

$$\forall t = 1 \ldots T, i = 1 \ldots I: \quad \boldsymbol{p}_{it}^{min} \leq \boldsymbol{p}_i^t \leq \boldsymbol{p}_{it}^{max} \tag{3.37}$$

$$\forall t = 1 \ldots T, j = 1 \ldots J: \quad 0 \leq \boldsymbol{w}_j^t \leq w_j^{max} \tag{3.38}$$

$$0 \leq \boldsymbol{s}_j^t \leq s_j^{max} \tag{3.39}$$

$$0 \leq \boldsymbol{l}_j^t \leq l_j^{max} \tag{3.40}$$

$$\boldsymbol{l}_j^t = \boldsymbol{l}_j^{t-1} + \eta_j \boldsymbol{w}_j^t - \boldsymbol{s}_j^t \tag{3.41}$$

$$\boldsymbol{l}_j^0 = l_j^{in}, \quad \boldsymbol{l}_j^T = l_j^{end}, \tag{3.42}$$

$$\forall t = 1 \ldots T: \quad \sum_{i=1}^{I} \boldsymbol{p}_i^t + \sum_{j=1}^{J} (\boldsymbol{s}_j^t - \boldsymbol{w}_j^t) \geq \boldsymbol{d}^t. \tag{3.43}$$

In models describing power generation systems, I denotes the number of thermal units, while J corresponds to the number of pumped hydro storage plants. These pumped storage plants are used to cut off the peaks of the load curve. The remaining curve has a more uniform structure, which means that the thermal units can operate within their favorable ranges. If $\upsilon_t(D)$ denotes the optimal value function of the following parametric problem:

Problem 3.6 (Parametric Thermal Dispatch)

$$\upsilon_t(D) := \min_{p} \sum_{i=1}^{I} c_i p_i$$

subject to:

$$p_{it}^{min} \leq p_i^t \leq p_{it}^{max}$$

$$\sum_{i=1}^{I} p_i \geq D$$

then the economic dispatch problem 3.5 can be rewritten omitting the thermal power variables p_i.

Problem 3.7 (Storage Problem with Hidden Units)

$$\min_{(\boldsymbol{s},\boldsymbol{w})} \mathbb{E} \sum_{t=1}^{T} \boldsymbol{v}_t \left(\boldsymbol{d}^t - \sum_{j=1}^{J} (\boldsymbol{s}_j^t - \boldsymbol{w}_j^t) \right) \tag{3.44}$$

subject to:

$$\forall t = 1 \ldots T, j = 1 \ldots J: \qquad 0 \leq \boldsymbol{w}_j^t \leq w_j^{max} \tag{3.45}$$

$$0 \leq \boldsymbol{s}_j^t \leq s_j^{max} \tag{3.46}$$

$$0 \leq \boldsymbol{l}_j^t \leq l_j^{max} \tag{3.47}$$

$$\boldsymbol{l}_j^t = \boldsymbol{l}_j^{t-1} + \eta_j \boldsymbol{w}_j^t - \boldsymbol{s}_j^t \tag{3.48}$$

$$\boldsymbol{l}_j^0 = l_j^{in}, \quad \boldsymbol{l}_j^T = l_j^{end}. \tag{3.49}$$

Only storage plants occur in this problem as in the problem 3.4. Moreover, the linearization of problem 3.7 coincides with problem 3.4 with the following replacement:

$$\boldsymbol{\lambda}_t = \frac{\delta}{\delta d} \boldsymbol{v}_t(d).$$

Usually, the function $\boldsymbol{v}$ as defined in problem 3.6 is not smooth, but it is appropriate to take a smooth approximation instead. Since the solution procedure for problem 3.6 is based on a priority list, the resulting piecewise linear function could be mollified as follows. The piecewise linear function $\boldsymbol{v}$ is given as:

$$\boldsymbol{v}(d) = \frac{d - d_{k+1}}{d_k - d_{k+1}} f_k + \frac{d - d_k}{d_{k+1} - d_k} f_{k+1} \tag{3.50}$$

if d resides in $[d_k, d_{k+1}]$. The table $\{(d_k, f_k)\}_{k=1}^{K}$ is built beforehand, and it contains the breakpoints of that function and their function values. Within the regions $\{[d_k - \delta, d_k + \delta]\}_{k=2}^{K-1}$, δ small enough, the function is replaced by a quadratic one, which builds a smooth connection of the segments $[d_{k-1}, d_k]$ and $[d_k, d_{k+1}]$.

In case of a smooth $\boldsymbol{v}$ function, descent directions of problem 3.4 are as well descent directions of the problem 3.7.

Thus, the following algorithm solves problem 3.5. First, the basic priority list for problem 3.6 is built. Then, an iteration method is applied to problem 3.5 with directions taken from problem 3.7, where the function $\boldsymbol{v}$ is replaced by a mollified version, and the mollifier region gets smaller ($\delta \to 0$) as the iteration goes on.

Problem 3.8 (Storage Problem of a Chained System)

$$\min_{\boldsymbol{x}} \mathbb{E} \sum_{t=1}^{T} \sum_{i=1}^{I} \boldsymbol{\lambda}^t \boldsymbol{x}_i^t \tag{3.51}$$

subject to:

$$\forall i = 1 \dots I,\ \forall t = 1 \dots T: \quad x_i^{min} \leq \boldsymbol{x}_i^t \leq x_i^{max} \tag{3.52}$$

$$\forall j = 1 \dots J,\ \forall t = 1 \dots T: \quad 0 \leq \boldsymbol{l}_j^t \leq l_j^{max} \tag{3.53}$$

$$\forall t = 1 \dots T: \quad \boldsymbol{l}^t = \boldsymbol{l}^{t-1} + M\boldsymbol{x}^t \tag{3.54}$$

$$\forall j = 1 \dots J: \quad \boldsymbol{l}_j^0 = l_j^{in}, \quad \boldsymbol{l}_j^T = l_j^{end}, \tag{3.55}$$

where M is the incidence matrix of the water network consisting of nodes (i.e. reservoirs l_j) and arcs (flows x_i).

Since the linkage (3.54) is no longer one-dimensional, the descent step algorithm needs some information about the direction, which has to be computed by a small linear program. This information has to be stored, such that it can be used for the update step. Because it is unlikely that the increased amount of implementational work is regained by a reasonable decrease of the computation time, the method of projected gradients (cf. [LP66] and [Pol97](textbook)) seems to be more appropriate.

3.1.5 Algorithm

Section 3.1.3 presented necessary conditions for the existence of a descent direction. These conditions depend on the binary variables b_k^{up} and b_k^{down}. However, the existence of a direction with the steepest descent can be verified without examining all possible combinations of the binary variables. First of all, the condition (3.15) does not depend on the b_k^{down}-, d_k^{down}-, and r_k^{down}-variables, while condition (3.16) does not depend on the b_k^{up}-, d_k^{up}-, and r_k^{up}-variables. For a direction of steepest descent with k as a root node the term $\lambda_k \pi_k + r_k^{up}$ has to be as small as possible. Since r_k^{up} is a recursively defined sum, each term can be minimized separately under the consideration of a non-zero step-length. The analysis leads to the following rules:

$$b_k^{up} = \begin{cases} 1 & \text{iff } \left(\min\{l^{max} - l_k, \min_{\kappa \in Succ(k)} d_\kappa^{up}\} = 0\right) \\ & \quad \vee \left(\lambda_k \pi_k \leq \sum_{\kappa \in Succ(k)} r_\kappa^{up}\right) \wedge (x_k - x^{min} > 0) \\ & \quad \vee (k \text{ is a leaf}) \\ 0 & \text{otherwise} \end{cases} \tag{3.56}$$

$$b_k^{down} = \begin{cases} 1 & \text{iff } \left(\min\{l_k, \min_{\kappa \in Succ(k)} d_\kappa^{down}\} = 0\right) \\ & \quad \vee \left(\lambda_k \pi_k \leq \sum_{\kappa \in Succ(k)} r_\kappa^{down}\right) \wedge (x^{max} - x_k > 0) \\ & \quad \vee (k \text{ is a leaf}) \\ 0 & \text{otherwise} \end{cases} \tag{3.57}$$

The fulfillment of condition (3.15) guarantees the existence of a direction with increased level realizing a lower objective value. To this direction a g-subtree corresponds, which starts at the node $k = ([\omega]_{t_0}, t_0)$. The leaves are the following nodes $\tilde{k} = ([\nu]_{t_1}, t_1)$, where $\nu \in [\omega]_{t_0}$ and t_1 is defined as $t_1 = \arg\min_{t \in [t_0+1,T]} \{t | b^{up}_{([\nu]_t, t)} = 1\}$. That means they are the first nodes in each scenario following the root and having the flag $b^{up}_{\tilde{k}}$ set. The same applies to the direction with decreased level. Here, $b^{up}_{\tilde{k}}$ is replaced by $b^{down}_{\tilde{k}}$.

Hence, descent steps can be found, if all auxiliary variables are recursively computed, and for each node the descent-step-conditions (3.15) and (3.16) are examined. If a descent step is found, an update of all intermediate nodes (between k and $\tilde{k}$) is performed.

Having these conditions in mind, the algorithm can be outlined as follows:

Step 1: Input and initialization;
Step 2: Determine a feasible point;
Step 3: Compute d^{up}_k, d^{down}_k, r^{up}_k, and r^{down}_k at all nodes;
Step 4: Search for the node (root of the g-subtree) with the steepest descent; unless it can be found, the current iterate is optimal $\rightarrow$ **STOP**;
Step 5: Update x_k and l_k at all nodes;
Step 6: Goto Step 3.

This descent algorithm *EXCHA* was implemented and tested. Each step of the algorithm requires only a few elementary computations, and in each step some variables attain upper or lower bounds. Hence, the algorithm is very efficient, as can also be seen in Figure 4.6, where the computing times in seconds of *EXCHA* on an HP-workstation are shown for a stochastic hydro storage problem with $T \leq 18$ and binary trees branching at all time periods with numbers of scenarios ranging up to 200.000. Notice that, in case a sequence of such problems with slightly different $\boldsymbol{\lambda}$-variables has to be solved, the last iterate of the previous problem can be used as a initial point for the next problem.

3.2 Dynamic Programming for Stochastic Programs

The occurrence of binary variables is quite often a difficulty very hard to overcome. Fortunately, in most cases the problem has a dynamic structure and satisfies the Bellman-Principle, see also [Bel57]. One of the well known applications of this principle is the Dijkstra algorithm for searching the shortest path in a graph. In this section an extension of this algorithm is presented. The stochastic nature requires the search of a tree with minimal expected length, which corresponds to a subtree of the scenario tree.

The **Bellman-Principle** is one of the basic principles in the optimization of a problem that has a dynamic structure. Next, this principle is explained on the shortest path problem.

Problem 3.9 (Shortest Path Problem) Let (V, E) denote a directed graph and c_e the length of arc e. The problem is to find a path connecting $\hat{x}$ and $\tilde{x}$, such that it's length is minimal, i.e.:

$$\min \left\{ \sum_{k=1}^{K-1} c_{(x_k, x_{k+1})} \;\middle|\; \exists\, K,\ \exists \{x_k\}_{k=1}^{K} \subseteq V,\ x_1 = \hat{x},\ x_K = \tilde{x} \right\}. \tag{3.58}$$

Theorem 3.3 (Bellman-Principle) In this theorem x_k denotes the k-th element of the sequence. Under the assumption that the sequence $\{\bar{x}_k\}_{k=1}^{K}$ is the optimal solution of the shortest path problem, $\Longrightarrow$ then, for any $k,\ 1 \leq k \leq K$ the sequences $\{\bar{x}_j\}_{j=1}^{k}$ are optimal for the problems

$$\min \left\{ \sum_{j=1}^{J-1} c_{(x_j, x_{j+1})} \;\middle|\; \exists\, J,\ \exists \{x_j\}_{j=1}^{J} \subseteq V,\ x_1 = \hat{x},\ x_J = \bar{x}_k \right\}.$$

And the remaining sequences, i.e. $\{\bar{x}_j\}_{j=k}^{K}$ for $k,\ 1 \leq k \leq K$, are optimal for the problems

$$\min \left\{ \sum_{j=1}^{J-1} c_{(x_j, x_{j+1})} \;\middle|\; \exists\, J,\ \exists \{x_j\}_{j=1}^{J} \subseteq V,\ x_1 = \bar{x}_k,\ x_J = \tilde{x} \right\}.$$

Proof: Since the problem is symmetric, the proof is only shown for the first point. Assuming that the sequence $\{\bar{x}_k\}_{k=1}^{K}$ is optimal for problem 3.9 and there exists a k^*, for which the sequence $\{\bar{x}_j\}_{j=1}^{k^*}$ is not optimal for the problem

$$\min \left\{ \sum_{j=1}^{J-1} c_{(x_j, x_{j+1})} \;\middle|\; \exists\, J,\ \exists \{x_j\}_{j=1}^{J} \subseteq V,\ x_1 = \hat{x},\ x_J = \bar{x}_{k^*} \right\}. \tag{3.59}$$

The optimality of $\{\bar{x}_k\}_{k=1}^{K}$ implies the boundness of (3.59), therefore an optimal solution $\{x'_i\}_{i=1}^{I}$ must exist, yielding

$$\sum_{i=1}^{I-1} c_{(x'_i, x'_{i+1})} < \sum_{j=1}^{k^*-1} c_{(\bar{x}_j, \bar{x}_{j+1})},\ x'_1 = \bar{x}_1, \text{, and } x'_I = \bar{x}_{k^*}. \tag{3.60}$$

Now, the sequence $\{x^\star_j\}_{j=1}^{J}$, $J := I + K - k^*$ constructed as

$$x^\star_j := \begin{cases} x'_j & \text{iff } j \leq I \\ \bar{x}_{j-I+k^*} & \text{otherwise} \end{cases}$$

has a lower value than the optimal value of (3.58), due to (3.60). Since the point $\{x^\star_j\}_{j=1}^{J}$ is feasible, this is a contradiction to the optimality of $\{\bar{x}_k\}_{k=1}^{K}$. Hence, the sequence $\{\bar{x}_j\}_{j=1}^{k^*}$ is optimal. #

Although the examples deals with a graph theoretical problem, many combinatorial problems with a separable objective function can be solved using this principle. The application of this principle to problems having a dynamic structure is called dynamic programming , see [Bel57, Ber87]. In this framework the path lengths are replaced by the cost-to-go functions.

Problem 3.10 The problem is to find a combination of $u \in \{0,1\}^T$ such, that it minimize a sum consisting of state dependent $(F(u^t))$ and transition dependent $(S^t(u^t, u^{t+1}))$ components, i.e.:

$$\min_u \sum_{t=1}^{T} F(u^t) + S^t(u^t, u^{t+1}), \tag{3.61}$$

where $S^T(u^T, .) \equiv 0$ for convenience.

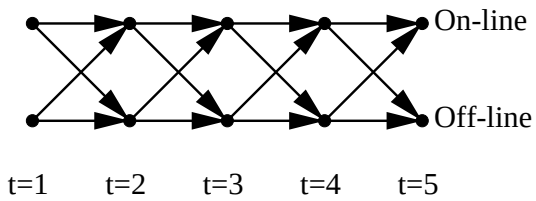

Figure 3.2: Deterministic Dynamic Programming

For $T = 5$ this problem is shown in Figure 3.2. The weights $F(u^t)$ are assigned to nodes, while the weights $S^t(u^t, u^{t+1})$ correspond to arcs. Then, the problem 3.10 is equivalent to the search of a shortest path, starting at $t = 1$ and ending at $t = T$. The cost-to-go function is defined as:

$$\gamma_t(u^t) := F(u^t) + \min_{\tilde{u}} \left\{ S^t(u^t, \tilde{u}) + \gamma_{t+1}(\tilde{u}) \right\}.$$

This is the so called "Backward formula" and the computation of problem 3.10 is obvious. One starts to compute the costs-to-go for $t = 5$, that means $\gamma_T(u^T) = F(u^T)$, and continues to compute these costs-to-go for $t = 4, 3, 2, 1$ carrying out a minimization in each step which is just the comparison of two values. The minimal value of $\gamma_1(u^1)$ and its accompanying trajectory provide the desired solution for (3.61).

In case of stochasticity the computation is similar:

Problem 3.11 Let $(\Omega, \mathcal{A}_T, \mathbb{P})$ be a probability space, $\zeta : \Omega \times \{1, \ldots, T\} \to \mathbb{R}^n$ is a random variable, and $\boldsymbol{\zeta}_t$ has to be measurable with respect to $\mathcal{A}_t$. The functions $F(u^t)$ and $S^t(u^t, u^{t+1})$ are the same as in problem 3.10 save for the occurrence of $\boldsymbol{\zeta}_t$, that means that these functions depend on the random variable. Then, the stochastic

dynamic problem reads:

$$\min_{\boldsymbol{u}} \mathbb{E} \sum_{t=1}^{T} F(\boldsymbol{u}^t, \boldsymbol{\zeta}_t) + S^t(\boldsymbol{u}^t, \boldsymbol{u}^{t+1}, \boldsymbol{\zeta}_t),$$

where $\boldsymbol{u}^t$ is measurable with respect to $\mathcal{A}_t$.

However, in the stochastic case the optimal solution does no longer correspond to a shortest path as in the deterministic case. In Figure 3.3 an example is shown, where $\boldsymbol{\zeta}_t$ is unique for $t = 1, 2, 3$ and has two different values for $t = 4, 5, 6$. The corresponding scenario tree is shown in the lower left corner, while the programming graph is displayed in the upper right corner. After the third time period the scenario tree branches.

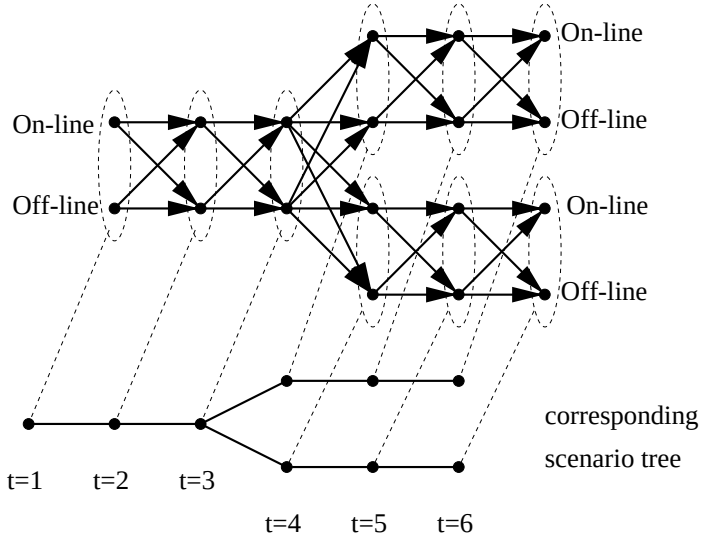

Figure 3.3: Stochastic Dynamic Programming

At this point the stochastic nature of the problem has to be taken into account. Instead of searching a shortest path the search is done for a tree with a minimal weighted length. Each branch of this tree corresponds to a scenario of the scenario tree. The function γ_t is now a random function and $\boldsymbol{u}$ a random variable. This is the only modification in comparison to the Dijkstra-algorithm. The cost-to-go function is defined as:

$$\gamma_t(u^t) := F(u^t) + \mathbb{E}\left(\min_{\tilde{u}} \left\{ S^t(u^t, \tilde{u}) + \gamma_{t+1}(\tilde{u}) \right\} \middle| \mathcal{A}_t \right). \tag{3.62}$$

Using this stochastic version of the "Backward Formula" the problem 3.11 is solvable with few elementary computations. In distinction to the deterministic problem a "Forward Formula" does not exists.

The same applies straightforward to problems with $u_t(\omega) \in \{0, 1, \ldots, N\}$ is — the formulae do not change.

Some combinatorial problems describing the operation of units include duration dependent startup costs. Duration dependent startup costs mean, that the cost of bringing a unit into operation depends on the duration that unit was offline before. Such a behavior occurs, if a unit cools down in the offline mode and has to be brought to a certain temperature within the startup procedure.

It is common in the unit-commitment-community to include this behavior into the dynamic programming model by arcs, which jump from a time period to a time period, which is not necessarily the next one.

Such a modeling is not possible, since the scenario tree might branch at intermediate time periods. Then, the long range jump arcs have to replaced by multi-arcs, i.e. arcs which are n-tuples of nodes (and $n > 2$). For such a generalized graph a theory of dynamic programming does not exists.

However, it is possible to include duration depending startup costs by extending the state space.

Chapter 4

Application to a Hydro-Thermal Power Generation System

The dynamic multi-stage stochastic model for the weekly cost-optimal generation of electric power by a generation system comprising thermal power plants and pumped hydro storage plants involves a large number of integer and continuous variables. Due to the huge size of the model, an application of state-of-the-art mixed-integer LP solvers is prevented. Therefore, a decomposition method is used in order to find suboptimal points. The stochastic Lagrangian relaxation is applied to the constraints that couple units. The dual problem, i.e. the maximization with respect to those Lagrange parameters, is solved by the bundle method presented in the next section.

4.1 Kiwiel's Proximal Bundle Method

There exists a wide variety of methods (cf. [HUL96a, HUL96b, Pol97]) that solve non-differentiable convex problems. Bundle methods seem to be the most efficient methods among them. Since they store information about the objective function during several iterations, they lead to a smaller number of function evaluation than other methods.

We consider the minimization of a convex function f on a nonempty closed convex set X in $\mathbb{R}^n$ and assume that the optimal set X_* is nonempty and we can compute $f(x)$ and a sub-gradient $g(x) \in \partial f(x)$ for each $x \in X$. The proximal bundle method [Kiw90, Kiw95] generates a sequence (x^k) in X converging to some element of X_*, and trial points $y^k \in X$ starting with $y^1 = x^1$ for evaluating sub-gradients $g(y^k)$ of f and its polyhedral lower approximation

$$\tilde{f}_k(x) = \max_{j \in J^k} \left\{ f(y^j) + < g(y^j), x - y^j > \right\} , \tag{4.1}$$

where J^k is a subset of $\{1, ..., k\}$ and $< ., . >$ denotes the scalar product in $\mathbb{R}^k$. In the iteration k the next trial point y^{k+1} is selected by

$$y^{k+1} \in \arg\min\{\tilde{f}_k(x) + \frac{1}{2}u_k\|x - x^k\|^2 : x \in X\} \tag{4.2}$$

where u_k is a proximity weight. A descent step to $x^{k+1} = y^{k+1}$ occurs if $f(y^{k+1}) \leq f(x^k) + \alpha v_k$, where $\alpha \in (0, 1)$ is fixed and $v_k = \tilde{f}_k(y^{k+1}) - f(x^k) \leq 0$. If $v_k = 0$, then x^k is optimal. Otherwise, a null step $x^{k+1} = x^k$ improves the next polyhedral function $\tilde{f}_k$. Strategies for updating u_k and choosing J^{k+1} are discussed in [Kiw90, Kiw95]. The method is implemented such that the cardinality of J^k is bounded (by some natural number NGRAD) and that it terminates if $-v_k$ is less than a given (relative) optimality tolerance.

This technique is applied to solve the dual stochastic problem (2.4) by putting $f = -D$ and $X = \mathbb{R}^{2N}_+$, where N is the number of nodes of the load scenario tree.

Our test runs confirmed the experiences of [Fel97] that a smaller number of function evaluations in case of a disaggregated objective function does not pay off the additional amount of work, which NOA 3.0 has to do in order to compute the next trial points. Hereby, the disaggregation of the objective function means that the bundle method stores information about function values and sub-gradients of several summands of the objective function separately. Additionally, the method has to deal with a far more complex bundle structure, and the solver for the quadratic problems needs more time for the computation.

Therefore, our computational experiments were made with the aggregated version and about 200 as the number of stored sub-gradients (NGRAD). Figure 4.1 shows a typical run. In the first few iteration the progress with respect to the objective value is quite big. After these first iterations the bundle method improves the approximation of the dual function by some trial points, as one can see in Figure 4.2. The objective function seems to be very flat around the optimal value, since the objective does not change much during the last iterations.

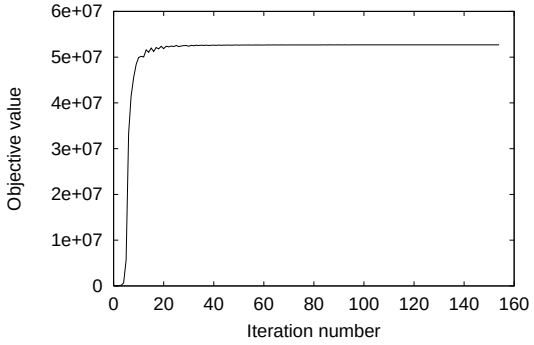

Figure 4.1: Objective function value during the *NOA 3.0* iterations

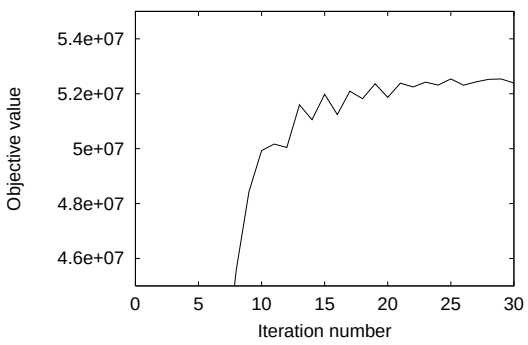

Figure 4.2: Details of Figure 4.1, showing the trial and real steps

Further experiments were made with different numbers of nodes of the scenario tree, i.e. with different dimension of the dual vector. For the dimensions used in our experiments (see Figure 4.3) the computation time depends almost linearly on the number of nodes.

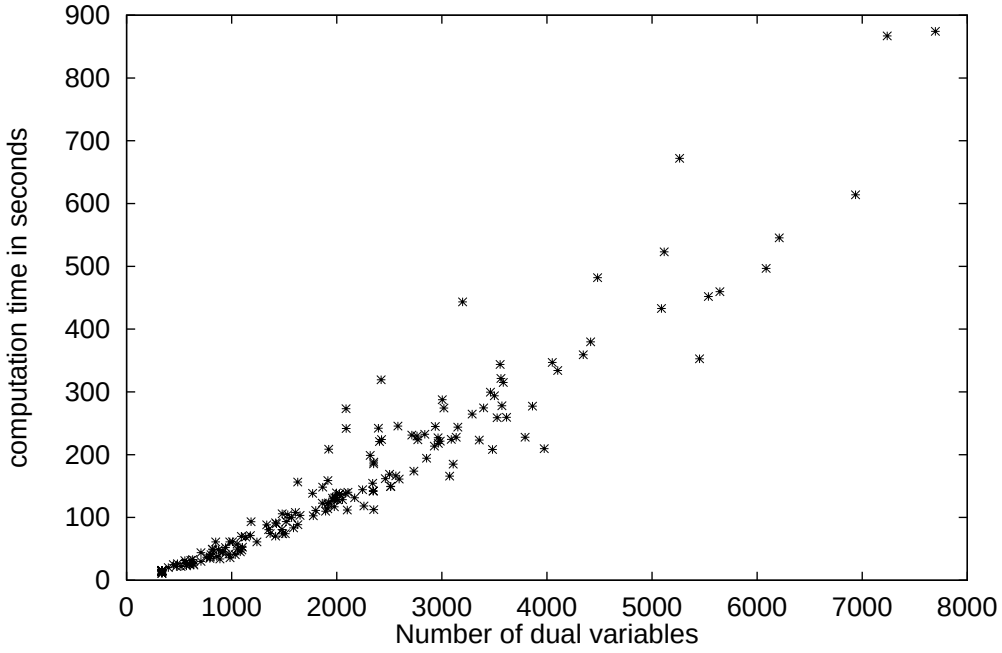

Figure 4.3: Computation times [s] of *NOA 3.0* on a HP-J280

Our computational experience with the proximal bundle code NOA 3.0 ([Kiw94]) for solving (2.4) was very encouraging (cf. Section 4). In our test runs, for instance, NOA 3.0 applied to solve (2.4) performed in 300 iterations as good as a standard sub-gradient method (with step lengths $\frac{1}{k}$) in 10.000 iterations.

The number of subgradients in the bundle increases with each iteration. This leads to a storage problem, if bundle elements are kept during all iterations. However, if elements are deleted during the iterations, cycling can occur. This did not happened in our numerical experiments.

Other methods like the reduced complexity bundle method ([LTZ98]) start with a new bundle each time a real step is performed. The bundle in this method does not contain the function value. Therefore, this bundle does not present a lower approximation of the non-differentiable function. Moreover, the bundle is used to determine a descent direction. Then, the real step includes a line-search.

Modified sub-gradient algorithms (cf. [RC99]) and reduced complexity bundle methods (cf. [LTZ98]) have their justification for problems, where function evaluations are cheap. Then, a large number of function evaluations is not a problem. These methods are also applicable, if a high precision is not required. Bundle

methods using lower approximations and an efficient method for the quadratic subproblems outperform these descent-step-like methods.

4.2 Descent Method for the Storage Subproblems

This method (cf. 3.1) is used for computing the single-unit subproblems of (2.4) that correspond to pumped hydro storage plants. The problem for the storage plant j reads:

Problem 4.1 (Storage Subproblem)

$$\tilde{D}_j(\boldsymbol{\lambda}) := \min_{(\boldsymbol{s}_j, \boldsymbol{w}_j)} \mathbb{E} \sum_{t=1}^{T} \boldsymbol{\lambda}^t \left(\boldsymbol{w}_j^t - \boldsymbol{s}_j^t \right) \tag{4.3}$$

subject to:

$$\begin{aligned} \forall t = 1 \ldots T: \quad 0 \leq \; \boldsymbol{w}_j^t \; &\leq w_j^{max} && (4.4)\\ 0 \leq \; \boldsymbol{s}_j^t \; &\leq s_j^{max} && (4.5)\\ 0 \leq \; \boldsymbol{l}_j^t \; &\leq l_j^{max} && (4.6)\\ \boldsymbol{l}_j^t \; &= \; \boldsymbol{l}_j^{t-1} + \eta_j \boldsymbol{w}_j^t - \boldsymbol{s}_j^t && (4.7) \end{aligned}$$

$$\boldsymbol{l}_j^0 = l_j^{in}, \quad \boldsymbol{l}_j^T = l_j^{end}, \tag{4.8}$$

where $\eta_j \in (0,1)$ denotes the cycle efficiency of the pumped storage plant j.

This problem is the same as problem 3.4 and solved by the algorithm described in Section 3.1.5. For further details of this method it is referred to [Now96].

This algorithm is used inside the *NOA 3.0*-bundle method. Therefore, the problem has to be solved again and again. Since it is an iteration method, it can make use of the optimal point of the previous iteration as a starting point.

Figures 4.4, 4.5, and 4.6 show numerical results for one pumped hydro unit and a small number of stages (i.e. less than 10), but with different number of scenarios on a HP-715 Workstation. The problems were randomly generated and solved using $\boldsymbol{s} = \boldsymbol{w} = 0$ as a starting point. The computing times in Figures 4.5 and 4.6 are the times, the program had been in memory. This explains the outliers in these figures. For a comparison with *MSLiP*(cf. [Gas90]) the reader is referred to [Now96].

Figure 4.4 shows that the number of descent steps is about half the number of scenarios. The reason for that is that in each step of the algorithm some variable attains an upper or lower bound and remains unchanged during the following iterations. Since each step requires only a few elementary computations, the

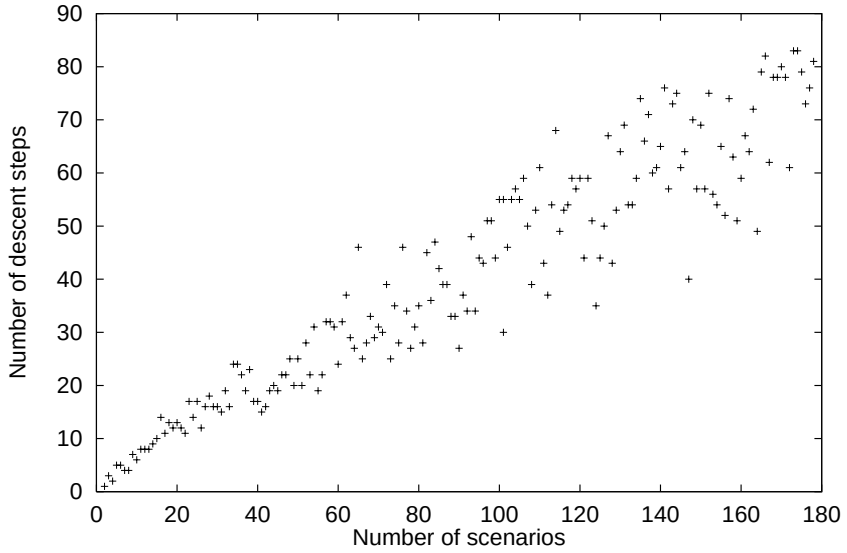

Figure 4.4: Number of steps of *EXCHA* against the number of scenarios

algorithms is very efficient as to be seen in Figure 4.5.

Further computations were made on a HP-735 Workstation with up to 30 stages and 220.000 scenarios. As Figure 4.6 shows, the required computation time grows linearly with the number of scenarios, hence linearly with the number of nodes.

4.3 Stochastic Dynamic Programming

The stochastic dynamic programming algorithm (cf. 3.2) is used to solve the single-unit subproblems, which correspond to thermal units:

Problem 4.2 (Thermal Subproblem)

$$\tilde{D}_i(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \min_{(\boldsymbol{u}_i, \boldsymbol{p}_i)} \mathbb{E} \sum_{t=1}^{T} FC_i(\boldsymbol{p}_i^t, \boldsymbol{u}_i^t) + SC_i^t(\boldsymbol{u}_i) \tag{4.9}$$

$$-(\boldsymbol{\lambda}^t - \boldsymbol{\mu}^t)\, \boldsymbol{p}_i^t - \boldsymbol{\mu}^t \boldsymbol{u}_i^t p_i^{max} \tag{4.10}$$

subject to:

$$\forall t = 1, \ldots, T \; : \; \boldsymbol{u}_i^t p_i^{min} \leq \boldsymbol{p}_i^t \leq \boldsymbol{u}_i^t p_i^{max} \,, \tag{4.11}$$

and $(\boldsymbol{u}_i^t, \boldsymbol{p}_i^t)$ are measurable with respect to $\mathcal{A}_t$.

This problem is a multi-stage stochastic mixed-integer problem and it seems, that it has the same complexity as the original problem 2.1. Fortunately, this problem

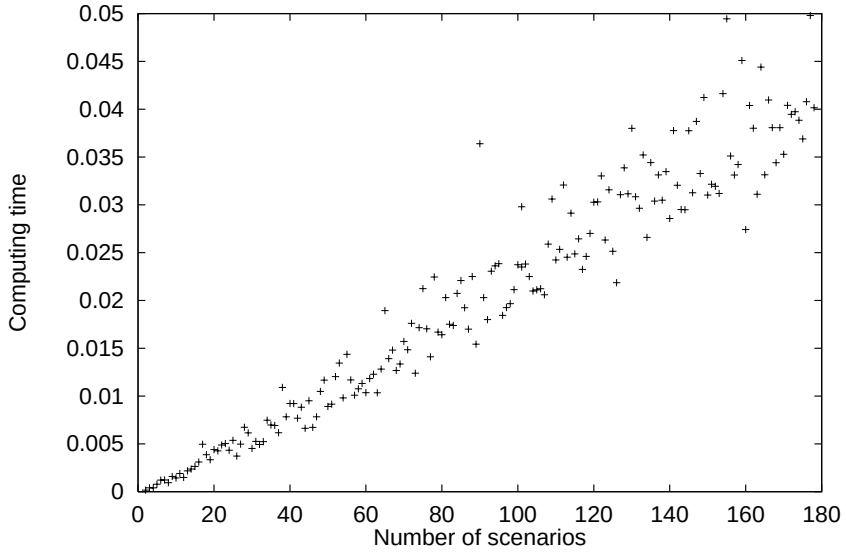

Figure 4.5: Computing time of *EXCHA* against the number of scenarios

has a usable structure — the formulation as a nested minimization problems paves the way for efficient methods.

Problem 4.3

$$\tilde{D}_i(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \min_{\boldsymbol{u}_i} \mathbb{E} \sum_{t=1}^{T} F_i(\boldsymbol{u}_i^t, \boldsymbol{\lambda}^t, \boldsymbol{\mu}^t) + SC_i^t(\boldsymbol{u}_i) - \boldsymbol{\mu}^t \boldsymbol{u}_i^t p_i^{max}\,, \tag{4.12}$$

where $F_i(\boldsymbol{u}_i^t, \boldsymbol{\lambda}^t, \boldsymbol{\mu}^t)$ is the result of the inner minimization:

$$F_i(u, \lambda, \mu) := \min_p \{FC_i(p, u) - (\lambda - \mu)p\} \tag{4.13}$$

subject to:

$$\forall t = 1, \ldots, T \;:\; up_i^{min} \leq p \leq up_i^{max}\,, \tag{4.14}$$

and $\boldsymbol{u}_i^t$ is measurable with respect to $\mathcal{A}_t$.

The inner minimization problem is one dimensional, and in case of piecewise linear or piecewise convex quadratic fuel cost functions FC_i the minimization can be carried out explicitly.

Hence, the remaining problem is of combinatorial nature. The start-up costs $SC_i^t(\boldsymbol{u}_i)$ depend on the components $\boldsymbol{u}_i^\tau$ of $\boldsymbol{u}_i$ for $\tau = t, t-1, \ldots, t-\tau_i^c$, where τ_i^c is the time the unit i needs to cool down. This is not suitable for dynamic programming, because these costs depend on more than 2 nodes. In order to apply the dynamic programming algorithm to stochastic programs, the state space is extended by including the recent history, such that the start-up costs depend

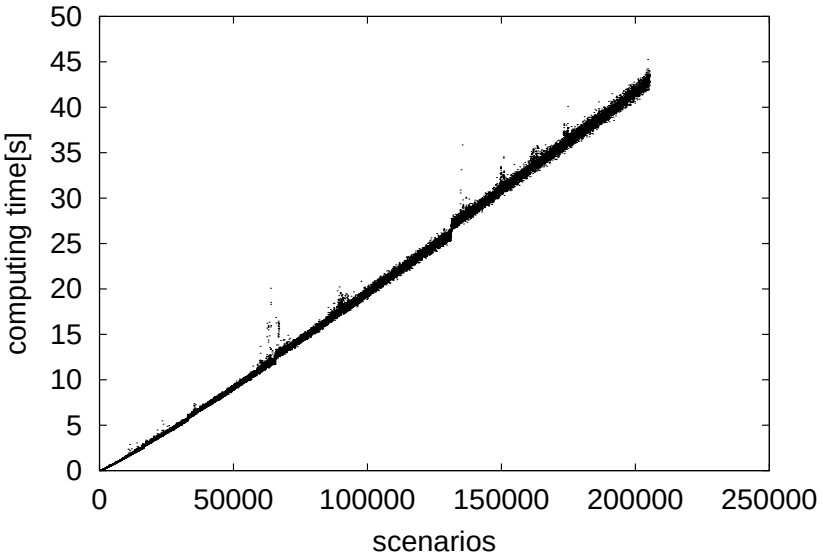

Figure 4.6: Computation times [s] of *EXCHA*

just on the current and the previous state. The stochastic dynamic programming graph (without weights) is defined as:

$$V_{SDP} := V \times (\{(1,\tau)|\tau = 1,\ldots,\tau_1^{max}\} \cup \{(0,\tau)|\tau = 1,\ldots,\tau_0^{max}\}) , \quad (4.15)$$

$$E_{SDP} \subseteq V_{SDP} \times V_{SDP} . \quad (4.16)$$

Constraints like minimum up- and down-times are imposed by limiting the feasible transactions. Figure 4.7 shows a part of the state transition graph of a thermal unit having a minimum up-time of 6 hours, a minimum down-time of 5 hours, and a cooling-down-time of 8 hours. These feasible transactions reduce the set of arcs E_{SDP}:

$$\forall e \in E_{SDP} : \Pi_{\mathcal{T}_\mathcal{R}}(e) \in \mathcal{T}_\mathcal{R} , \quad (4.17)$$

where $\Pi_{\mathcal{T}_\mathcal{R}}$ is the projection on the transaction graph, defined as:

$$\Pi_{\mathcal{T}_\mathcal{R}}(e) := \{ ((u,\tau),(u',\tau'))|\, e = ((k,u,\tau),(k',u',\tau'))\} \quad (4.18)$$

An example with 14 time periods and 3 scenarios for a unit with a (4, 3, 5) configuration is shown in Figure 4.8. The corresponding formulae to (3.62) for the cost-to-go functions $\gamma_i(n)$ read:

$$\gamma_i(n) := \bar{F}_i(n) + \mathbb{E}\left(\min_{(n,\tilde{n})\in E_{SDP}} \bar{S}(n,\tilde{n}) + \gamma_i(\tilde{n}) \middle| n \right) , \quad (4.19)$$

where $n, \tilde{n}$ are nodes of the dynamic programming graph. Each node n is a tuple $(([\omega]_t, t), u, \tau)$, and τ denotes the duration the unit was in state u_i^t.

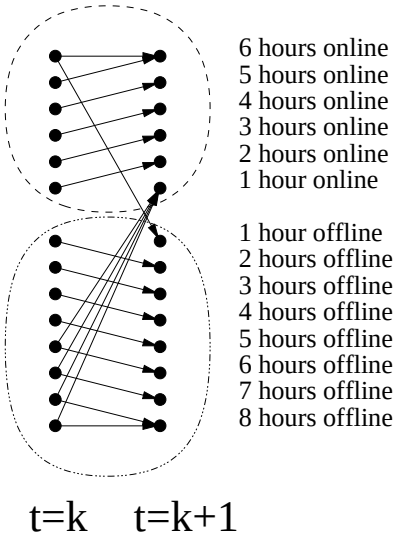

Figure 4.7: Feasible transactions $\mathcal{T}_{\mathcal{R}^-}$ an example

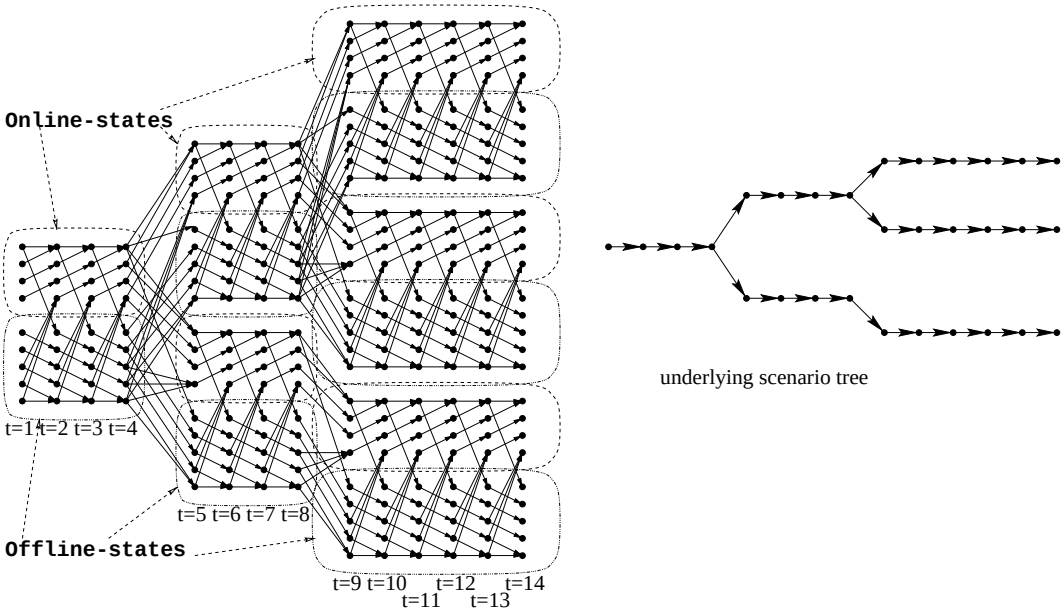

Figure 4.8: The complete dynamic programming graph for a unit with 4 hours must-on time, 3 hours must-off time, and 5 hours cooling time.

The node weights $\bar{F}_i(n)$ are defined as

$$\bar{F}_i\left(([\omega]_t, t), u, \tau\right) := \begin{cases} F_i(1, \lambda_t(\omega), \mu_t(\omega)) & \text{iff } u = 1\,, \\ 0 & \text{otherwise}\ . \end{cases} \tag{4.20}$$

The startup costs are $\boldsymbol{\lambda}, \boldsymbol{\mu}$ independent:

$$\bar{S}(n, \tilde{n}) := \begin{cases} \alpha_i + \beta_i(1 - e^{-\frac{\tau}{\kappa_i}}) & \text{iff } u = 0 \wedge \tilde{u} = 1 \\ 0 & \text{otherwise}\ , \end{cases} \tag{4.21}$$

where $n = (([\omega]_t, t), u, \tau)$ and $\tilde{n} = (([\omega]_{t+1}, t+1), \tilde{u}, \tilde{\tau})$ for simplicity, and $(\alpha_i, \beta_i, \kappa_i)$ denote the startup cost parameters for unit i. Figure 4.9 shows an example. Here,

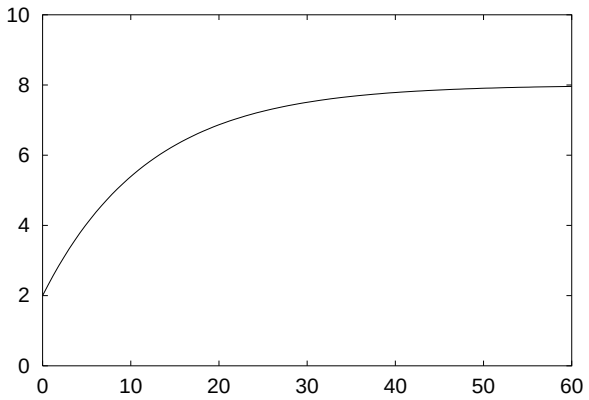

Figure 4.9: The exponential startup curve

8 is the cost if the unit was off-line for a very long time, so everything had to be warmed up. If it was down for just 10 hours, there are some parts, which still have a higher temperature.

> There is an opportunity to reduce the startup costs, when the unit is off-line for a few hours. This method is called "Banking" and is not considered in this paper. Banking means, that the thermal plant uses fuel to keep its operational temperature without producing electricity.

Since it is very sophisticated to distinguish slightly different startup costs, the unit is considered to be cold, if it was off-line for about 30 hours. In the VEAG-system a 5% accuracy leads to a 40 hours for large coal fired thermal plants, 8 hours for medium size plants and 1 hour for gas turbines. Our computational results showed that the number of nodes in the dynamic programming graph is not a limiting factor. From experience the finer approximation of startup costs leads to a speed up of the bundle method. The explanation might be a smoother dual function.

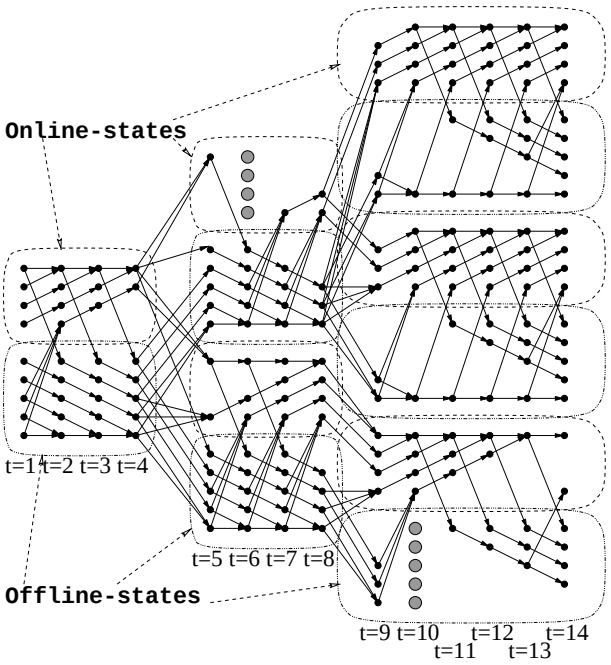

Figure 4.10: The reduced dynamic programming graph

The dynamic programming algorithm is capable to consider must-on and must-off constraints too. This is done by reducing the dynamic programming graph. In order to get a unit online at a certain time period of a certain scenario, the corresponding off-line nodes are deleted. In Figure 4.10 these nodes are marked gray. The deletion of these nodes causes the deletion of other nodes, which do not have a ancestor or a successor. In Figure 4.10 the node for $t = 4$ and a online time of 1 hour is not feasible due to the must-off constraint at time $t = 6$ of the upper scenario. This cause the non-existence of the node $t = 6$ and 3 hours online for all scenarios. That means, that must-up and must-off constraints of one scenario have impacts on the parts of other scenarios. Hence, the must-on and must-off constraints imply further restrictions on the stochastic dynamic programming graph in order to get feasible schedules.

Definition 4.1 A stochastic dynamic programming graph is irreducible, if all nodes belong to at least one feasible schedule. That means, for all nodes $\tilde{v}$ there exists a schedule s such that $\tilde{v}$ belongs to a dynamic programming path, which corresponds to that schedule, i.e.:

$$\forall \tilde{v} \in V_{SDP}\,, \exists s : V \to V_{SDP}\,, \tilde{v} \in \mathsf{Im}(s)\,, \tag{4.22}$$

$$\forall (v, v') \in E \;:\; (s(v), s(v')) \in E_{SDP}\,. \tag{4.23}$$

The following set $G^\star$ denotes the set of nodes to which a feasible schedule corresponds, i.e.:

$$G^\star := \max \left\{ \tilde{v} \in V_{red} \;\middle|\; \begin{array}{c} \forall (v,v'),(v',v) \in E\,, \exists \tilde{v}' \in G^\star \\ \Pi_V(\tilde{v}) = v, \Pi_V(\tilde{v}') = v', \\ ((\tilde{v}',\tilde{v}) \in E_{SDP} \vee (\tilde{v},\tilde{v}') \in E_{SDP}) \end{array} \right\}, \tag{4.24}$$

where Π_V denotes the Cartesian projection onto V, and V_{red} the nodes of the stochastic dynamic programming graph, which do not conflict with must-on and must-off constraints. The definition is correct, because $G^\star \subseteq V_{SDP}$, and because of the fact that the union of these sets fulfills the right hand side, whenever each of the two sets fulfills the right hand side. The computation of such an irreducible set of nodes is done by the alternating application of the following 2 operators:

$$P(G) := G \setminus \{\tilde{v} \,|\, \neg\exists \tilde{v}' \in G \,:\, (\Pi_V(\tilde{v}'), \Pi_V(\tilde{v})) \in E\} \tag{4.25}$$

$$N(G) := G \setminus \{\tilde{v} \,|\, \exists e = (\Pi_V(\tilde{v}), v') \in E, \forall \tilde{v}' \in G \,:\, \Pi_V(\tilde{v}') \neq v'\}\,. \tag{4.26}$$

The first operator deletes all nodes, which do not have an ancestor in G, the latter deletes nodes without a successor for at least one branch of the scenario tree. The procedure starts with $G^0 = V_{red}$ and a decreasing sequence is built by $G^{k+1} := PNG^k$.

Corollary 4.1 The sequence built as described above has the following properties:

- There exists a k_0 such that $G^{k+1} = G^k$ is valid for all $k \geq k_0$.
- $G^\star = \lim_{k\to\infty}(PN)^k(V_{red})$

Proof: The first item is true, since the application of the operators creates a decreasing sequence of set; because the first set G^0 had a finite number of elements, the method has to stop after a finite number of steps.

Due to the definition of P and N the following inclusions are fulfilled:

$$\forall G \supseteq G^\star \,:\, G^\star \subseteq P(G),\, G^\star \subseteq N(G)\,. \tag{4.27}$$

Therefore, $G^\star \subseteq \lim_{k\to\infty}(PN)^k(V_{red})$ is fulfilled. The finiteness of steps gives $PNG^{k_0} = G^{k_0}$ That means, all elements of G^{k_0} do not fulfill the remove-conditions of (4.25) and (4.26). Hence, all elements fulfill the conditions of (4.24). Since $G^\star$ is the maximum of such sets, the sets $G^\star$ and G^{k_0} are equal. #

First, the nodes V_{SDP} are created and the arcs are added that correspond to feasible transactions as in $\mathcal{T}_\mathcal{R}$. During the following arcs are removed if a neighboring node is deleted. Next, nodes violating the must-on and must-off constraints are removed. After that the iteration procedure can start.

At the beginning all nodes are marked black. Nodes corresponding to $t = 1$ are marked green. Next, all nodes are considered — if they have a green ancestor,

they are also marked green. This corresponds to the application of the operator P. Green nodes for $t = T$ are marked red. Next, nodes are marked red, if they have at least one red node for each branch of the scenario tree. This corresponds to the application of the operator N. All black and green nodes are removed, the remaining red nodes are marked black. If at least one node was removed, the procedure starts with marking the start nodes ($t = 1$) green. After a finite number of iterations no more nodes are deleted and the procedure stops. This technique is also known as coloring. The remaining graph contains feasible schedules only. If the graph is empty, then the must-on and must-off constraints conflict and a feasible schedule does not exists.

After setting up the stochastic dynamic programming graph, arc weights are calculated (i.e. the $SC_i^t(u_i)$) and assigned.

In each iteration of the dual method, when the stochastic dynamic programming part is called, the minimization in $\tilde{F}_i(n)$ is carried out and assigned to the nodes as weights. Then, the calculations of the cost-to-go values is done using the backward-formula, and the paths yielding the minimum are marked. The forward run along the marked paths gives the schedules with minimal costs.

On a HP-J280 such a dynamic programming step for 1 unit, 168 time periods and 1 scenario is done within 4 ms.

4.4 Lagrangian Reduction and Facet Search

The solution of the dual problem gives optimal dual parameters within a certain accuracy. Since there is no way to use dual information in order to obtain primal optimal points, heuristics as developed in Section 2.6 are used instead.

The problem, a good primal solution is searched for, is the VEAG-problem 2.1, i.e.:

$$\min_{(\boldsymbol{u},\boldsymbol{p},\boldsymbol{w},\boldsymbol{s})} \mathbb{E} \sum_{i=1}^{I} \sum_{t=1}^{T} FC_i(\boldsymbol{p}_i^t, \boldsymbol{u}_i^t) + SC_i^t(\boldsymbol{u}_i), \tag{4.28}$$

subject to several constraints. This objective function is modified to an extended-real valued function by including the constraints. The min-max-problem with the Lagrange-function reads:

$$\min_{(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w})} \max_{(\boldsymbol{\lambda},\boldsymbol{\mu})} L(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w};\boldsymbol{\lambda},\boldsymbol{\mu})\,. \tag{4.29}$$

This formulation ensures, that all constraints are met, when the optimal value of this problem is less than infinity. One can think of a relaxation of the following type:

$$\min_{(\boldsymbol{s},\boldsymbol{w})} \max_{(\boldsymbol{\lambda},\boldsymbol{\mu})} \min_{(\boldsymbol{u},\boldsymbol{p})} L(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w};\boldsymbol{\lambda},\boldsymbol{\mu})\,. \tag{4.30}$$

From a mathematical point of view this problem has some interesting properties. The inner *max-min*-problem is the same as the

dual problem to the generation problem of a system comprising thermal units only. The use of pumped hydro storage plants results in load curve, where the peaks and the off-peaks are reduced. If this relaxed problem could be solved and optimal $\boldsymbol{s}, \boldsymbol{w}$ variables could be obtained, then fast heuristics (like [ZG88]) for the purely thermal problem could be applied for getting primal points $\boldsymbol{u}, \boldsymbol{p}$, which yield an objective value close to the optimum.

However, the nested *min-max-min* sequence inhibits some computational difficulties. Without more sophisticated techniques three different algorithms are required. The nesting of the min and max operators results in a nesting of the algorithms — and therefore in an increased time complexity of the compound algorithm.

The further relaxation, i.e.:

$$\max_{(\boldsymbol{\lambda},\boldsymbol{\mu})} \min_{(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w})} L(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w};\boldsymbol{\lambda},\boldsymbol{\mu})\,, \tag{4.31}$$

does not increase the duality gap, because

$$D(\boldsymbol{s},\boldsymbol{w};\boldsymbol{\lambda},\boldsymbol{\mu}) := \min_{(\boldsymbol{u},\boldsymbol{p})} L(\boldsymbol{u},\boldsymbol{p},\boldsymbol{s},\boldsymbol{w};\boldsymbol{\lambda},\boldsymbol{\mu}) \tag{4.32}$$

is a convex function in $(\boldsymbol{s},\boldsymbol{w})$, and therefore the strong duality holds. This close relationship gives rise to the assumption that the dependency of the binary values $\boldsymbol{u}$ on the dual values $(\boldsymbol{\lambda},\boldsymbol{\mu})$ in (4.30) is of the same type as the dependency of b on λ in (2.78). Under normal circumstances, the optimal output level of a unit is maximal, if the unit is online (see (4.13) and (4.14)). Thus, the problem appears to be a binary one.

In order to investigate the dependency of $\boldsymbol{u}$ on $(\boldsymbol{\lambda},\boldsymbol{\mu})$, the number of $\boldsymbol{u}$-components were counted for different values of $(\boldsymbol{\lambda},\boldsymbol{\mu})$. Since $(\boldsymbol{\lambda},\boldsymbol{\mu})$ is an element of a high dimensional space, a certain direction was chosen. In [ZG88] it was shown, that $(0,\mathbb{1})$ is a direction, where thermal units tend to be online. Because our reserve constraints were formulated differently, the direction $(\mathbb{1},\mathbb{1})$ was taken instead.

In Figures 4.11 – 4.14 the dotted line shows the behavior for dual values very close to the dual optimum — the dual problem was solved to a high accuracy with high computational effort. The solid lines in these figures shows, that the ramp of the dotted line is approximated as the dual values draw near to the optimal values. The ordinate value 0 denote the dual values as obtained by the the dual solver. Positive values denote a perturbation in the direction $(\mathbb{1},\mathbb{1})$, while negative values denote a perturbation in the opposite direction.

Figure 4.11 shows the dependency for inaccurate dual points. At least 2000 binary variables have wrong values. Thus, few iterations of the solver of the dual problem do not provide the requested accuracy. The first origin of the predicted

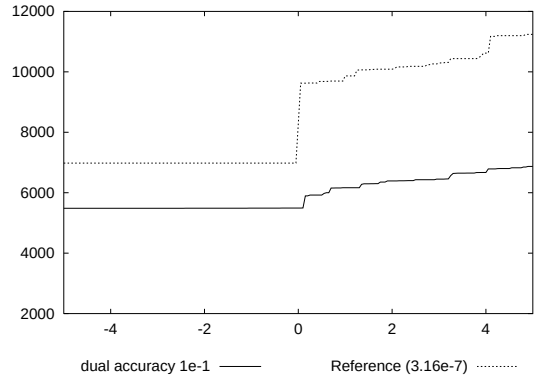

Figure 4.11: Dependency at 10^{-1}.

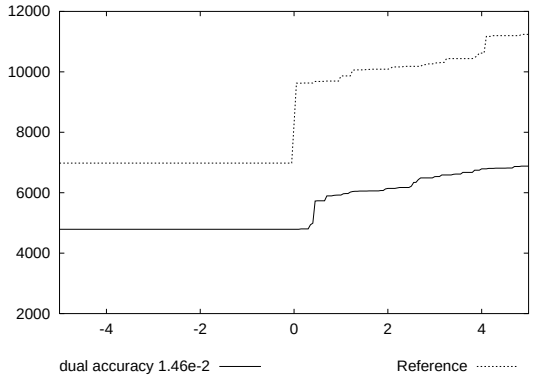

Figure 4.12: First Ramp at 1.45×10^{-2}.

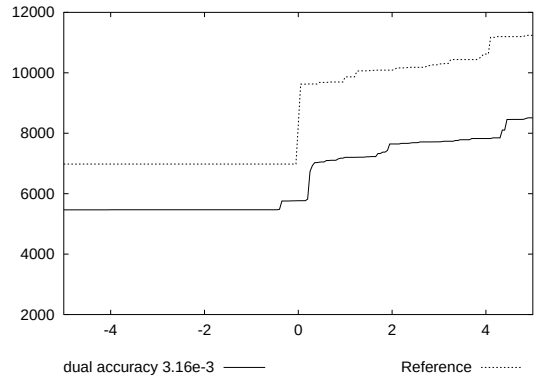

Figure 4.13: Stair at 3.16×10^{-3}.

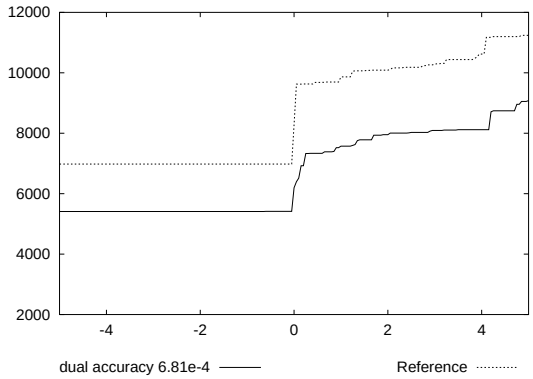

Figure 4.14: Ramp at 6.81×10^{-4}.

ramp shows Figure 4.12. During the dual iterations the dependency approaches the ramp, see Figures 4.13 and 4.14. These pictures (4.11 – 4.14) give a hint that the dual problems should be solved with at least 0.5×10^{-4} accuracy in order to get a good characterization of the binary values.

As assumed, the ramp shows that many binary values may change their values, if the dual parameter is perturbed slightly as predicted in theorem 2.2. Even there is a ramp, many binary variables are fixed. Thus, the binary variables belong to 3 different groups:

1. variables, which stay zero,
2. variables, which are always one,
3. variables, which take different values.

This distinction forms the basis for the heuristics. As mentioned in Section 2.6, the dual information results in schedules, which are preferred by the inner minimization problem 4.2. In fact, the dual information recommends to have units with high operational costs offline, while cheap units are suggested to be online in all time periods. Figure 4.15 shows the preferred schedule of a unit

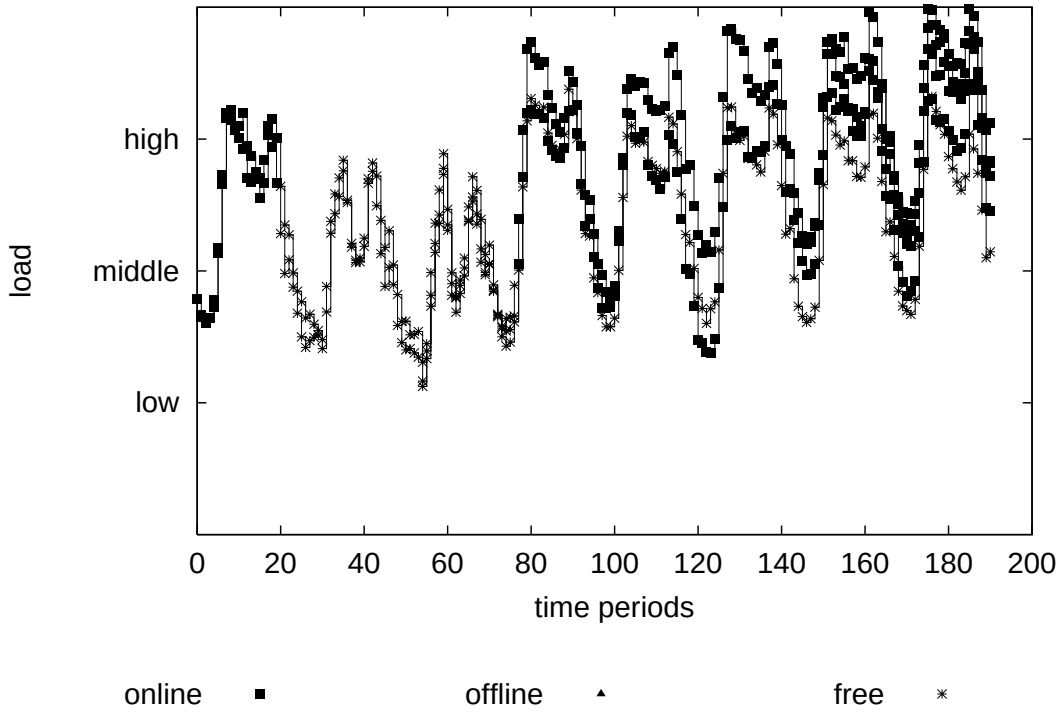

Figure 4.15: Cycle Unit in a high load week starting on Friday morning

with medium operational costs. The demand of that week is more than the average. Therefore, the online state of a unit with medium costs is preferred. Only in scenarios with a lower demand and during the weekend (time periods 20 – 80) the unit is neither preferred to be online nor to be offline. Therefore, the schedule of this unit during these time periods should be settled together with the schedules of the other units of the system in order to reach low overall operation costs.

Figure 4.16 shows the opposite. The optimal Lagrange parameters of this low load example lead to another set of recommended schedules. During the weekend (time periods 65 – 115) and in some scenarios with a lower demand the offline state of this unit is preferred. In all other time periods the schedule of this unit has to be settled in coordination with the other units of the system.

Hence, the heuristics starts with fixing the binary variables, which are preferred by the dual variables. This reduces the dimension of the problem[1]. Then, there is only a few percentages of the binary variables left for the optimization. However, the number of combinations is still to big for a complete enumeration.

The free binary variables form a hypercube in a certain high dimensional space. Two corners of this hypercube are certainly not optimal, the corner with all components equal to zero and the corner with all components equal to one.

[1]Therfore, I call it Lagrangian reduction.

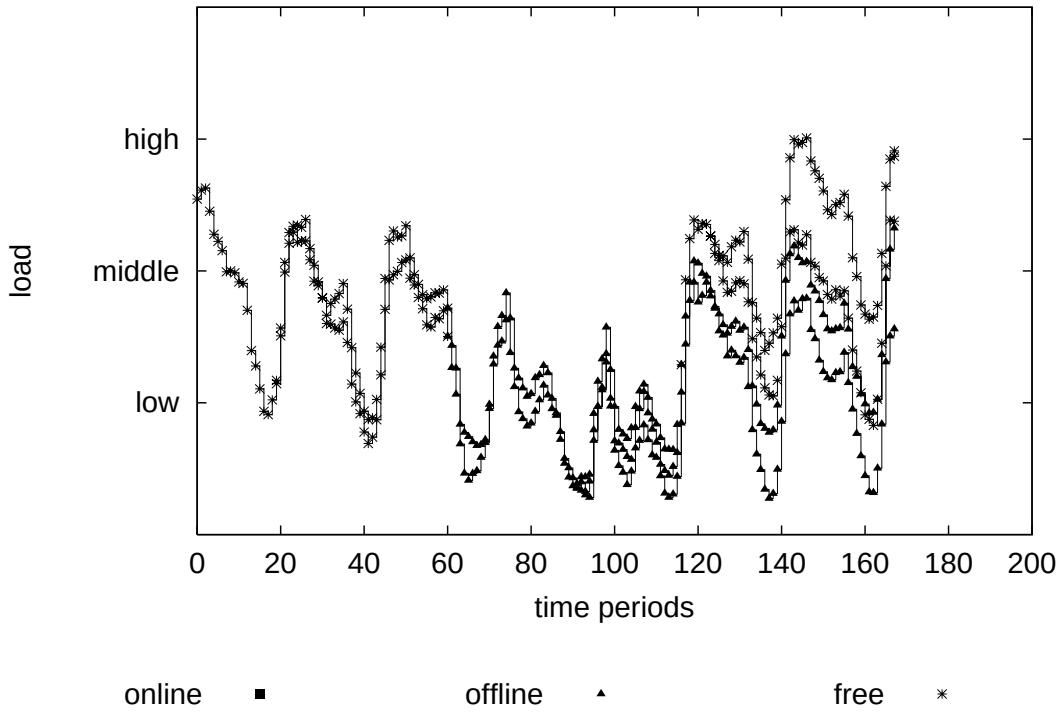

Figure 4.16: Cycle Unit in a low load week starting on Wednesdays noon

Moreover, the first one is not even feasible in most cases. If all undecided units are online, then the sum of fixed costs is unnecessarily high. Therefore, the corner with the best objective value must lie on a path from the first to the latter corner. The remaining question is the path to follow. Since the Lagrangian multiplier do not supply information about that point, a greedy algorithm is applied that uses the information obtained during the search.

The heuristics can be outlined as follows:

1. Solve the dual problem.
2. Detect the preferred schedules and fix the corresponding binary variables.
3. Set all undecided variables to zero.
4. Solve the corresponding economic dispatch problem.
5. Check feasibility of the problem:
   - problem is infeasible: Select a time period, where the additional amount needed to satisfy the reserve constraint is maximal. Choose an undecided unit and set it online in this and the neighboring time periods that are requested by dynamic programming.

- problem is feasible:
  - If the objective value is the best so far, then store the schedule and the output levels.
  - Select a time period with units operating within the high price regions of their cost functions. Choose an undecided unit and set it online in this and the neighboring time periods that are requested by dynamic programming.

6. If there was no undecided unit in the previous step, keep that time period out of consideration and select another time period. If there is no time period and no undecided unit, then **STOP**

7. **GOTO** 4.

Note, in each step an economic dispatch problem has to be solved. But the sequence was constructed in a way, that subsequent schedules differ just in few binary variables. That means, the next schedule comprises one more online unit. This unit could be used to increase the outcome of the thermal part and this leads to a reduced turbine or an increased pumping mode of the hydro part. Thus, the optimal dispatch of one schedule is a good starting point for the iteration method that solves the next schedule.

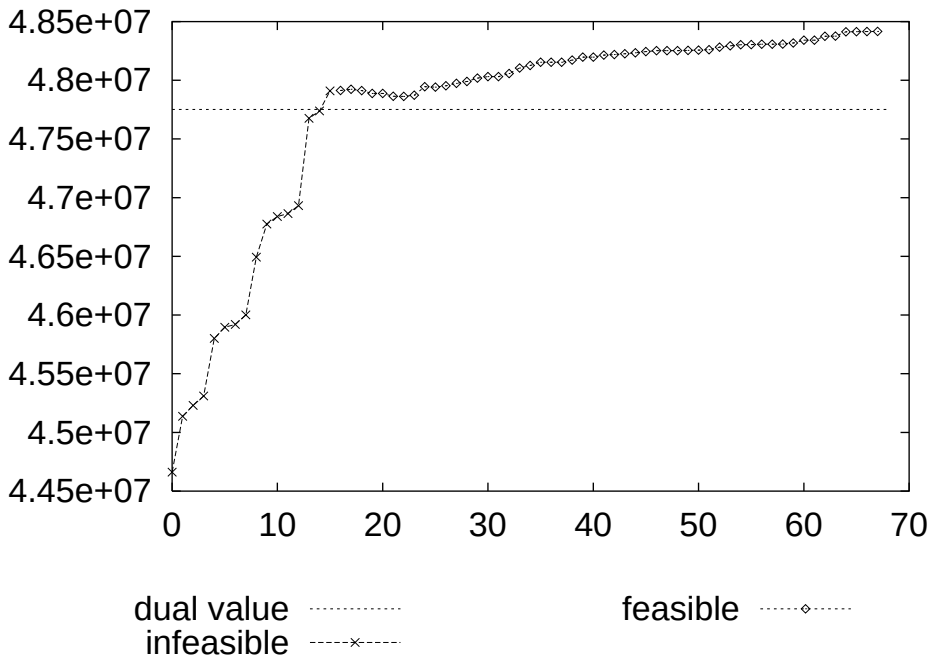

Figure 4.17: Objective value and feasibility of the chosen sequence

In Figure 4.17 the objective values and the feasibilities to the economic dispatch problems of the sequence are displayed. At the beginning (0) of the se-

quence the objective value is quite small due to the small number of units being online. Because the number of online units is to small, the problem is not feasible. Therefore, the objective value, being less than the value of the dual problem, forms not a contradiction to the duality theory. As long as units are brought online, the fixed operation costs are increased. At a certain point of the sequence (e.g. at 13 in Figure 4.17) the number of online unit is sufficiently large to meet the demand. However, in most cases this was only possible by the use of pumped hydro storage plants to a great extent. But the excessive use of pumped storage plants is not optimal. When further thermal units are brought into operation, the operation of the pumped storage plants and their operational (and hidden) costs can be reduced. Figure 4.17 shows, that the distance to the value of the dual problem can be cut by half.

At each step there is an increase of the fix costs of the thermal units, which is compensated by the decrease of the operational costs of the hydro plants during the first steps — until a favorable operation of the pumped storage hydro units is reached. Then, bringing further units online leads to a net increase of the objective function.

Figure 4.17 also shows that once the feasibility is reached all further points are feasible too. This gives rise to the opposite direction: the sequence starts at the point with all undecided units online. In each step an undecided unit is switched off in a time period with superfluous reserve and in neighboring time periods due to dynamic programming. This approach allows to abort the sequence evaluation at the first time period with infeasibility without cutting off the best point. This is also justified by corollary 2.1.

4.5 Comparison with the modified Zhuang/Galiana Heuristics

> The heuristics developed in the previous section belongs to the greedy algorithms. It is in the nature of greedy algorithms to fail at some problems. Therefore an excessive numerical experiment was done in order to compare the facet search with a well known heuristics from [ZG88].

The original Zhuang/Galiana heuristics (cf. [ZG88]) was developed for a system comprising thermal units only. This search method for a "reserve feasible solution" (RFS-procedure) modifies the current schedule in the following way:

1. Solve the dual problem in the same manner as in Section 2.3. The reserve constraint should look like:

$$\forall t = 1, \ldots, T \; : \; \sum_{i=1}^{I} u_i^t p_i^{\max} \geq d^t + r^t \, . \tag{4.33}$$

2. Select a time period t with a maximal violation of the reserve constraint. If no reserve constraint was violated, then the current schedule is optimal and **STOP**

3. For each thermal unit i compute the increase Δ_i of μ^t, which is necessary to get unit i online in that time period due to the dynamic programming algorithm.

4. Increase μ^t by the smallest[2] amount of Δ_i.

5. Compute the new schedule by dynamic programming.

6. **GOTO** 2.

The reserve constraint in form of (4.33) ensures the fulfillment of the demand constraint with appropriate output levels if the reserve constraint is fulfilled. On the base of the schedule found by the heuristics a economic dispatch can be performed such, that the sum of fuel and startup costs is minimal with respect to the fixed schedule.

But the VEAG-system comprises pumped hydro storage plants too. Therefore, the operation of the hydro plants has to be fixed in order to apply the RFS-procedure of (cf. [ZG88]). The optimal operation of the system found by primal methods (cf. [DGM$^+$97]) gives a hint, how the hydro plant operation should look like.

The pumped storage plants are used in peaks to reduce the output of the thermal system, while in off-peaks the turbined amount of water has to be pumped uphill again, which results in a increased output of the thermal system during off-peaks periods. The numerical experiments shows that pumped hydro storage plants are usually used to cut off peaks and off-peaks of the demand curve.

Such a method, which also takes the unsatisfied reserve into account, is described in detail in [GMR$^+$97, DGM$^+$97]. This greedy heuristics reads:

1. Initialize the hydro plants in order to satisfy the balance equations.

2. Find a time period, where the reserve constraints is violated or if such a time period does not exists, a time period with high operational costs.

3. Find a suitable time period with satisfied reserve and low operational costs.

4. If such a pair of time periods can't be found, then **STOP**.

5. Determine the amount of energy/water to be exchanged.

6. **GOTO** 2.

[2] + small ϵ for numerical stability

The key of this WRS[3] heuristics is the pairing of time periods. In peak periods energy is used which was stored in a off-peak period. Thus, there is a flow of energy from off peak periods to peak periods with losses (limited cycle efficiency) and flow constraints (limits on engines, pumps, and the reservoir level).

The basic flow equation corresponding to the nodes prevents the application to a stochastic problem, the comparisons were made on deterministic problems, i.e. problems with one scenario.

Because we were interested in the average performance, a huge number of problems were generated and the results are plotted in Figures 4.18 and 4.19. In order to support the visualization the line $y = x$ was added to the plots.

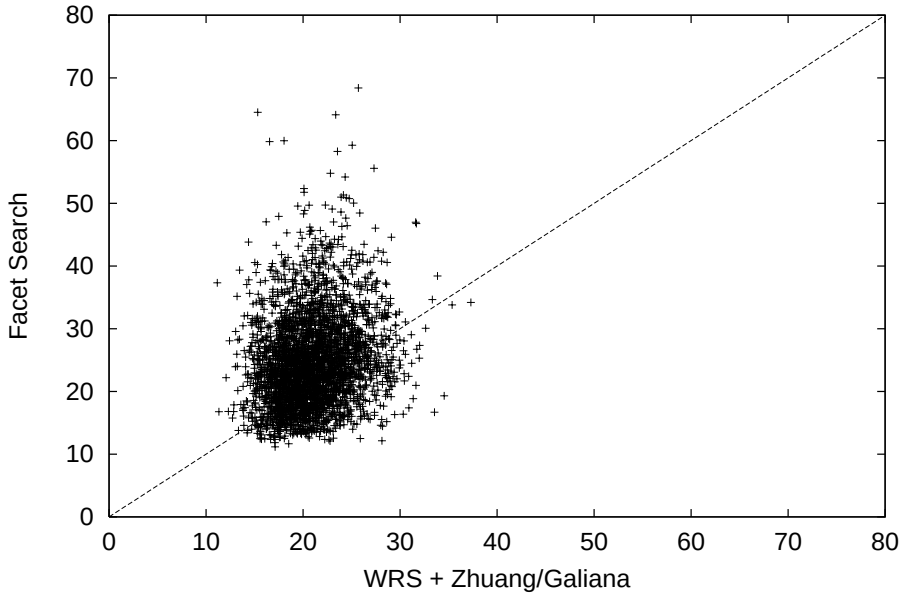

Figure 4.18: Comparison of computing times

In Figure 4.18 the computing time of both methods are plotted. With few exceptions the modified Zhuang/Galiana heuristics needs not more than 30 seconds for the problems, while the facet search in average needs 5 seconds more. Quite often more time is spend in the facet search. One reason for that behavior might be the variation of the length of the sequence that has to be evaluated.

In Figure 4.19 the upper bound for the duality gaps are plotted. The upper bounds are given in percent and denote the relative difference between the found primal value and the dual value, while the term duality gap refers to the relative difference of the "optimal" primal value and the dual value.

The tail to the right denotes experiments, where the facet search led to an upper bound of less than 0.4%, but the modified Zhuang/Galiana heuristics gave

[3]Water ReScheduling

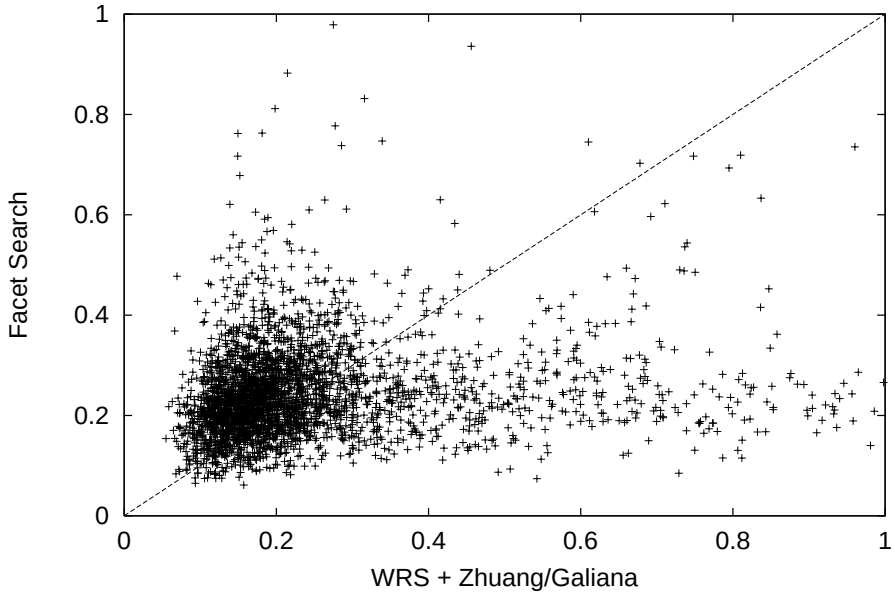

Figure 4.19: Comparison of upper bounds for the duality gaps

results with up to 1% upper bound. Furthermore, most problem were solved with a bound of less than 0.35% by the facet search. If the problems were solved by the WRS-Z/G heuristics, bounds of more than 0.35% were obtained for an essential part of the problems.

The second solution method (WRS+Z/G) combines 2 heuristics. The first one has no information about the thermal system. It just tries to reduce the gap between peak load and off-peak load. But the operation of the pumped storage hydro plants should reduce unnecessary state changes of the thermal system. Sometimes these two aims do not lead to the same schedule and this WRS heuristics will fail.

The second heuristics looks for a schedule of the thermal system, which satisfies the reserve constraints. At this time, the opportunity to change the hydro schedule accordingly, is not used. Obviously, this is a lack of the second methodology. On the other hand, the heuristics deal with problems of lower complexity. This explains, why the WRS+Z/G heuristics is faster in average.

4.6 Economic Dispatch

The economic dispatch problem is the same problem as the problem 2.1 save for the fact that the $\boldsymbol{u}$'s are fixed. In case of piecewise linear fuel cost functions FC_i this problem is a stochastic multi-stage linear problem. Problems of that type

can be solved by methods like MSLiP (cf. [Gas90]). Another opportunity is to solve the deterministic equivalent by efficient methods like CPLEX.

Fortunately, the economic dispatch problem inhibits the same structure as the problem 3.5. Therefore, some effort was made in order to apply the same technology of Section 3.1.

First, the problem 2.1 has to be transformed into a storage problem with hidden thermal units like problem 3.7. This transformation uses a parametric thermal dispatch. At this time, the additional reserve constraint (2.28) has to be taken into account. Hence, the parametric problems reads:

$$v_{(r,u)}(d) := \min_{p_i} \sum_{\substack{i=1 \\ u_i=1}}^{I} FC_i(1, p_i) \tag{4.34}$$

subject to:

$$\forall i = 1, \ldots, I, u_i = 1 \quad : \quad p_i^{min} \leq p_i \leq p_i^{max} \tag{4.35}$$

$$\sum_{\substack{i=1 \\ u_i=1}}^{I} p_i \geq d \tag{4.36}$$

$$\sum_{\substack{i=1 \\ u_i=1}}^{I} p_i \leq \sum_{\substack{i=1 \\ u_i=1}}^{I} p_i^{max} - r\,. \tag{4.37}$$

The solution technique for this problem is based on a priority list. This lists tells, in which order the units should increase their output level, if the output of the system does not meet the demand. The list contains the segments of all fuel costs functions, which were assumed to be piecewise linear. Let c_k denote the slope of the segment k, which has the length d_k and belongs to unit i_k. Assume further, that the list is ordered, i.e. $c_0 \leq c_1 \leq \ldots \leq c_{K-1} \leq c_K$. Then the algorithm reads:

1. Initialize
$$\forall i = 1, \ldots, I \; : \; p_i = \begin{cases} p_i^{min} & u_i = 1 \\ 0 & u_i = 0 \end{cases}, \tag{4.38}$$
set $k = 0$.

2. If $\sum_{i=1}^{I} \geq d$ then **STOP**, p is optimal.

3. If $u_{i_k} = 0$,
then set $k := k + 1$ and **GOTO** 2
else

   - $\Delta_k := \min\{d_k, d - \sum_{i=1}^{I} p_i\}$

- increase p_{i_k}, i.e. $p_{i_k} := p_{i_k} + \Delta_k$.
- if $\sum_{i=1}^{I} p_i = d$ then
  - set $k^\star := k$,
  - $d^{up} := d_k - \Delta_k$
  - $d^{down} := \Delta_k$
  - $k^{++} := \min\{k | k > k^\star \wedge u_{i_k} = 1\}$
  - $k^{--} := \max\{k | k < k^\star \wedge u_{i_k} = 1\}$
- set $k := k + 1$.

4. **GOTO** 2.

This algorithm shows, that the same priority list can be used even for different values of $\boldsymbol{u}$. Note, the feasibility of the problem has to be checked separately. In addition, more information like objective value and distances to the neighboring segments can be derived easily:

- objective value: $v_{(r,u)}(d) = \sum_{k=1}^{K} c_k \Delta_k$,
- distance to the next segment or the end of the feasibility region:

$$d^{\uparrow} := \min\{d^{up}, \sum_{\substack{i=1 \\ u_i=1}}^{I} p_i^{max} - r - \sum_{i=1}^{I} p_i\},$$

- distance to the previous segment: $d^{\downarrow} := d^{down}$,
- the slope of the current, the previous, and the next segment: $c_{k^\star}$, $c_{k^{--}}$, and $c_{k^{++}}$.

Problem 4.4 (VEAG-Problem with Hidden Units) The remaining problem reads:

$$\min_{(\boldsymbol{s},\boldsymbol{w})} \mathbb{E} \sum_{t=1}^{T} v_{(\boldsymbol{r}^t,\boldsymbol{u}^t)} \left(\boldsymbol{d}^t - \sum_{j=1}^{J} (\boldsymbol{s}_j^t - \boldsymbol{w}_j^t) \right) \tag{4.39}$$

subject to:

$$\forall t = 1 \ldots T, j = 1 \ldots J: \qquad 0 \leq \boldsymbol{w}_j^t \leq w_j^{max} \tag{4.40}$$

$$0 \leq \boldsymbol{s}_j^t \leq s_j^{max} \tag{4.41}$$

$$0 \leq \boldsymbol{l}_j^t \leq l_j^{max} \tag{4.42}$$

$$\boldsymbol{l}_j^t = \boldsymbol{l}_j^{t-1} + \eta_j \boldsymbol{w}_j^t - \boldsymbol{s}_j^t \tag{4.43}$$

$$\boldsymbol{l}_j^0 = l_j^{in}, \quad \boldsymbol{l}_j^T = l_j^{end}. \tag{4.44}$$

This problem, comprising pumped storage plants only, can be solved by the descent method for storage problems presented in Section 3.1. At the point, where the step length is calculated, the distances to the neighboring segments are taken into account. The details are omitted here, because they do not contribute to the understanding of the method.

Another, already mentioned, opportunity to solve this problem is an LP-solver. Since we were interested in the performance of the adapted algorithm, this algorithm was compared with the commercial solver CPLEX.

The first experiment was made with an example with 25 thermal units, 8 pumped hydro storage plants, 40 time periods, and 1 scenario. The led to an LP with 2952 columns, 3720 rows and 9624 nonzero elements of the constraint matrix.

CPLEX-Function	Pricing-Strategies primal/dual					
	-1	0	1	2	3	4
simplex/primal	60.79	63.83	91.05	129.92	124.39	77.71
simplex/dual		50.82	46.56	51.79	65.69	51.35
baropt	16.0					
hybbaropt/primal	18.41	18.31	18.31	33.2	18.3	18.23
hybbaropt/dual		18.82	18.38	41.42	74.49	18.3
hybnetopt/primal	55.53	56.17	76.92	115.44	105.82	59.04
hybnetopt/dual		62.9	62.42	62.73	75.86	62.96

Table 4.1: Computation time in seconds of CPLEX for the small example

The computation time of the adapted algorithm was 1.85 seconds. The commercial solver CPLEX offers a lot of different algorithms and options for solving LP problems. Table 4.1 shows the corresponding computation times. The pricing strategies are options for the primal and the dual simplex solver. This table shows, that the best method within CPLEX for the economic dispatch problem is the barrier method. The simplex methods take at least twice as long, whatever options are used.

The small example gave a first hint that the barrier method would be the appropriate CPLEX solver to the economic dispatch problem. Table 4.2 shows the computation times of an example with the same configuration, but 192 time periods. This corresponds to an LP-problem with 14 200 columns, 17 856 rows and 46 256 non-zeros of the constraint matrix. The ECDISP algorithm took 50.95 seconds to solve the problem. Again, the barrier method was the best CPLEX-method — however, the adapted method was still faster than the general method for LP-problems.

The tables 4.1 and 4.2 justify that the adapted method should be compared with the barrier method. The results of further experiments are displayed in table 4.3. In the first column the number of scenarios are located. They correspond to

CPLEX-Function	Pricing-Strategies primal/dual					
	-1	0	1	2	3	4
Simplex/primal	1232.47	1188.4	1918.15	2664.14	2440.7	1696.9
Simplex/dual		1086.18	946.24	1103.48	1466.54	1083.8
baropt	94.78					
hybbaropt/primal	114.71	114.32	114.36	486.55	114.45	114.35
hybbaropt/dual		115.08	114.69	693.03	1424.86	114.84
hybnetopt/primal	957.66	910.39	1298.03	2252.83	1960.93	1162.68
hybnetopt/dual		1393.82	1253.76	1412.06	1833.96	1392.3

Table 4.2: Computation time in seconds of CPLEX for the bigger example

Scen.	Nodes	ct(ECDISP)	ct(CPLEX)	adv
1	168	6.96	46.31	6.65
2	252	11.02	64.35	5.84
3	336	18.69	97.61	5.22
4	392	41.25	150.37	3.65
5	462	29.48	162.47	5.51
6	504	37.17	228.76	6.15
7	588	47.93	206.00	4.30
8	630	40.06	249.56	6.23
9	687	43.09	305.43	7.09
10	723	61.59	289.79	4.71
11	792	67.17	500.30	7.45
12	828	95.57	356.88	3.73
13	930	86.73	461.54	5.32
14	966	116.81	534.78	4.58
15	1035	98.04	569.18	5.81
16	1071	107.79	529.14	4.91
17	1036	117.42	620.65	5.29
18	1064	103.33	504.41	4.88
19	1120	91.63	1720.33	18.77
20	1148	111.31	768.21	6.90
21	1232	131.94	243.27	1.84
22	1260	128.18	794.93	6.20

Table 4.3: Ratio of the computation times of CPLEX to ECDISP

a certain number of nodes of the scenario tree that are to be found in the second column. The next 2 column are the computation times of ECDISP and CPLEX. The last column displays the ratio of the computation time of CPLEX to that of ECDISP. It seems that ECDISP is about 5 to 6 times faster than the barrier method. The computations were done on a SUN workstation with 160 MByte memory and CPLEX 4.0 was used. It was not possible at this time to extend this comparison to examples with more than 22 scenario, because CPLEX ran out of memory. Figures 4.20 and 4.21 shows the numbers for test runs with ECDISP alone.

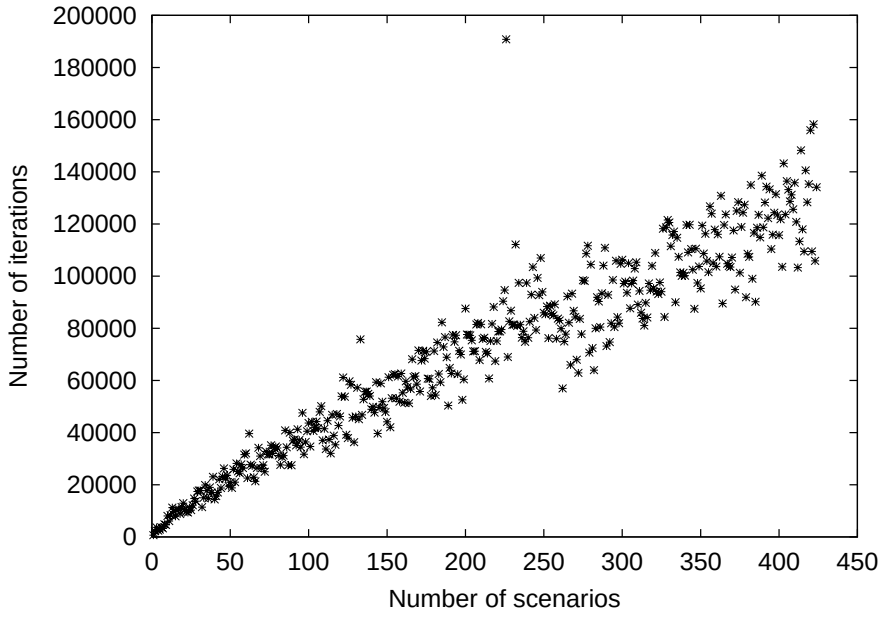

Figure 4.20: Number of ECDISP iterations against the number of scenarios

The number of iterations grows almost linearly with the number of scenarios. This can be explained by the fact, that the ECDISP method is a descent method. The number of facets, which a descent method has to cross, grows with the dimension. Since the scenarios are loosely coupled, i.e. at 1 node of the scenario tree, the number of facets grows almost linearly.

The same applies to the computation time. Figure 4.21 shows, that the computation time grows almost linearly with the number of scenarios.

4.7 Numerical Results

The solvers for subproblem discussed in the previous sections are combined to a compound algorithm, which can solve the problem 2.1 up to nearly optimality, and in addition can provide a stochastic solution for that problem.

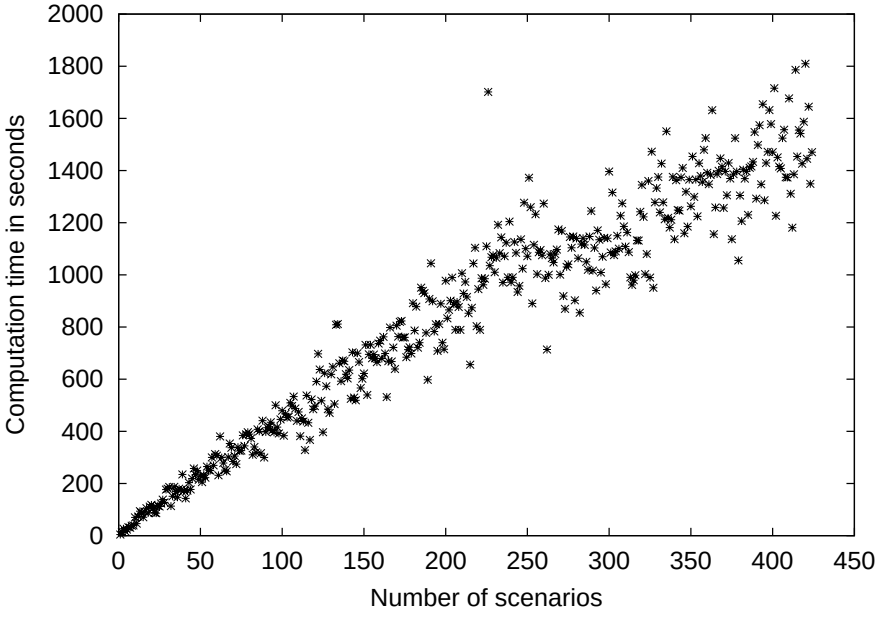

Figure 4.21: Computation times against the number of scenarios for the ECDISP algorithm

The data interface for the program are ASCII-files — the input files describe the configuration of the hydro-thermal power generation system and the load forecast, while the output files contain the schedules of all units and hydro plants.

The flow chart 4.22 shows in which order the sub-algorithms are executed. The upper part contains a cycle with an exit point, which is denoted by the question " Dual optimality reached?". Here, the dual problem 2.4 is solved with a certain accuracy. The computation time of this part is denoted by ct(NOA) in table 4.7.

> The FORTRAN-routines of NOA 3.0 were encapsulated into C++-classes. The user-supplied function *usefun*, used in NOA 3.0 for function evaluations, wraps the FORTRAN-call into calls of C-functions. This approach made it possible to use FORTRAN for the computation of the dual function and C++ for the solutions procedures of the subproblems, which must have access to data stored in C++-structures. In distinction to the suggested use of NOA 3.0 (cf. [Kiw94]) the package was used as a subroutine instead of as a program, because the maximization of the dual function is just a part of the compound algorithm. Since speed was one of the major aims, the problem setup was done by a C++-part and a slightly modified subroutine of the FORTRAN-package was called with prepared arguments.
>
> The descent method for the storage subproblems runs directly on the global data structure, which had suitable structure for this

algorithm. In each call by the usefun-function of NOA 3.0 the descent method starts with the last point of the previous iteration as a starting point. During the last iterations of the dual algorithm the Lagrange parameters do not change much. This leads to a small number of iterations of the descent method, since the last point of the previous iterations seems to be a good approximation of the optimal solution.

The stochastic dynamic programming algorithm traverses the scenario tree and the corresponding dynamic programming graph in various ways. It turned out, that the pointer structure of the scenario tree is not suitable for the forward and backward iterations of the dynamic programming algorithm. Here, a separate data structure was used instead of in-place-operations. The initialization of the stochastic dynamic programming structure contains the following steps:

1. Setup of a full stochastic dynamic programming graph, which bases on the scenario tree. This results in a graph like in Figure 4.8.
2. Marking forbidden nodes as dead nodes, like the marked nodes in Figure 4.10.
3. Coloring the graph as described in Section 4.3 in order to find the irreducible graph.
4. The stochastic dynamic programming graph is reorganized such, that there exist just nodes of the irreducible graph.
5. Since online nodes of the programming graph that correspond to the same node of the scenario tree have a common node weight, a separate data structure containing the node weights was added.
6. Arc weights are the startup costs. These costs do not depend on the dual parameters. Therefore, these weight are assigned to the arcs during the initialization.
7. Data structures, used to support the forward and backward iterations, and storage for the flags and auxiliary values for the stochastic dynamic programming algorithm are added.

In each stochastic dynamic programming call, the dual values are translated into node weights and stored into the corresponding places. Then, the basic stochastic dynamic programming algorithm is called. This routine ends with optimal values stored in the auxiliary data structures. They are evaluated and the corresponding results are stored within the global data structure.

The observation was made, that a common data structure for all algorithms leads to access problems. In former implementations of the dynamic programming algorithm it happens, that several values were

only accessible by indirect links, i.e. $c[i[n]] = a[b[n]] + d[k[n]]$. Our experience gave, that in increase of computation time for copying data is paid off by a more effective access to data within the sub-algorithm. In our implementation the stochastic dynamic programming does not essentially contribute to the computation time.

The next part forms the stochastic Lagrangian relaxation. This methods uses a perturbation analysis in order to fix as many binary variables as possible. The implementation follows exactly the description given in Section 4.4.

The lower part describes the facet search method. Here, the solver for the economic dispatch algorithm is called for each examined schedule. As proposed at the end of Section 4.4 the sequence starts with as many units online as possible. Note, during this facet search all units/time-periods remain unchanged, once they are fixed during the Lagrangian reduction.

# of sc.	nodes	ct(NOA)	ct($\sum$)	bound for d. gap
1	168	10.14	16.38	0.203
2	193	14.31	25.44	0.179
2	269	29.34	40.14	0.387
3	276	25.64	62.76	0.106
4	275	30.64	53.26	0.331
5	542	64.73	101.41	0.193
8	739	80.81	180.60	0.246
10	983	127.45	229.56	0.712
13	900	111.28	2281.31	1.009
17	1786	277.84	733.36	0.446
21	2098	350.84	530.64	0.389
24	2175	373.75	694.7	0.833
27	2208	379.63	8349.38	0.729
31	2558	523.10	9791.48	1.033
32	2173	359.14	3337.28	0.657
33	2241	482.03	4074.91	0.345
34	3043	496.72	1498.72	0.948
37	2823	459.60	7776.40	1.013
39	3848	874.25	4091.57	0.818

Table 4.4: Results for the compound algorithm, computation times (ct) are in seconds, ct($\sum$) denotes the computation time for the whole algorithm

Table 4.7 shows the computation times for test problems with different numbers of scenarios. The second column shows the number of nodes within the scenario tree. The dimension of the dual problem is twice this number. All dual problems were solved using 200 subgradients and an optimality tolerance of 10^{-5}. The third column shows the computation time for the dual problem, while the

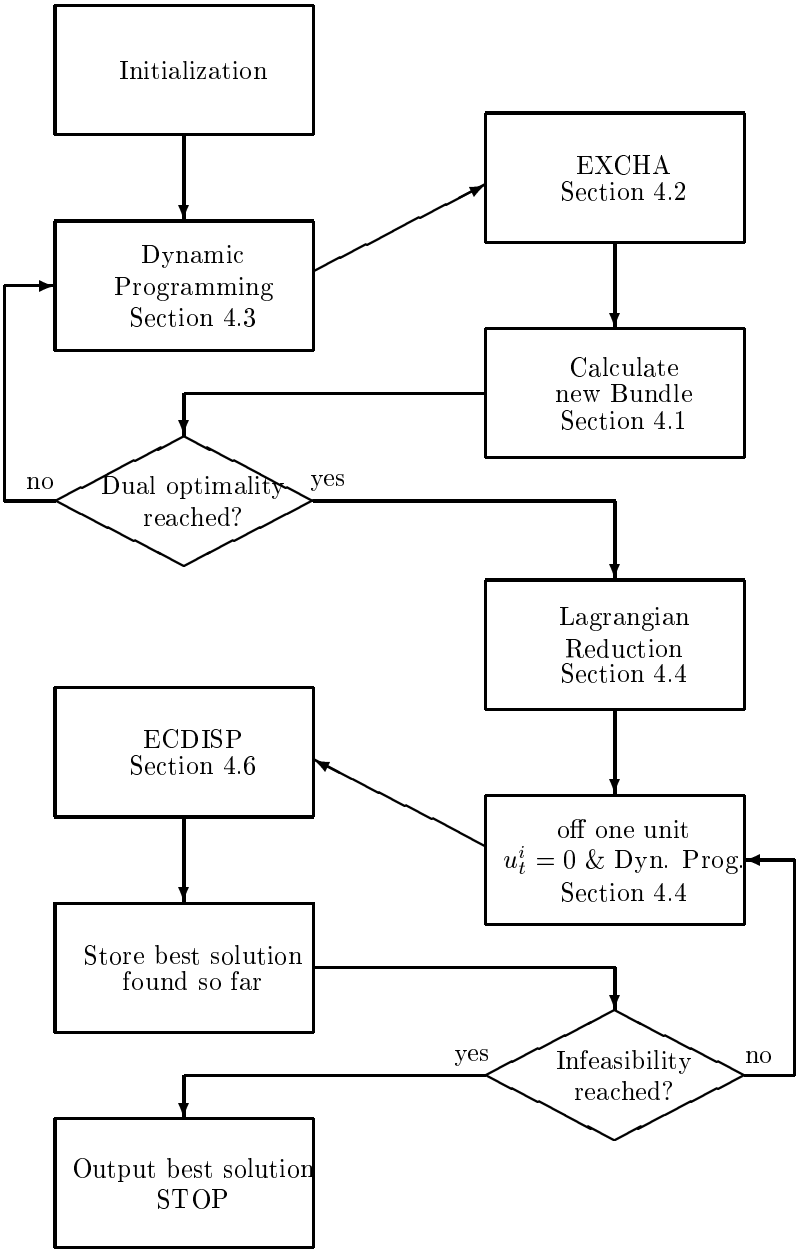

Figure 4.22: Flow Chart for the Complete Algorithm

next column shows the time consumption for the compound algorithm. The last column shows the computed bound for the duality gap. This bound is computed as:

$$b := \frac{v(\hat{x}) - v_D}{v_D}, \tag{4.45}$$

where $v(\hat{x})$ denotes the primal value of best solution, and v_D denotes the optimal value of dual problem. The dual gap itself is

$$g := \frac{v_P - v_D}{v_D}, \tag{4.46}$$

where v_P denotes the value of the optimal primal solution. Since exact method for the primal stochastic problem are not known, the gap cannot be determined. However, one is interested in the gap between the found primal solution and the optimal solution. Again, the proposed method provides an upper bound:

$$\frac{v(\hat{x}) - v_P}{v_P} \leq \frac{v(\hat{x}) - v_D}{v_D}. \tag{4.47}$$

Thus, the last column of table 4.7 certifies, that the best solution found during the facet search is at least 0.5% close to the optimum. Assuming, that there exists a gap between the optimal primal value and the optimal dual value, small upper bounds for the gap give rise to the assumption, that the facet search has found the optimal solution[4].

Some examples of table 4.7 show a larger bound for the duality gap. This fact coincides with a high amount of computation time. Numerical experiments have shown, that the number of economic dispatch problems, which are solved during the facet search of these problems, is very high. This is possible only, if the perturbation analysis within the Lagrangian reduction could not determine a sufficiently large number of fixed binary variable. Then, there are too many variables free for the facet search, which leads to a long sequence and to high computation times. Since the success of the Lagrangian reduction method depends heavily on the accuracy of the dual parameters, the problems should be solved again with a higher accuracy for the dual part.

Figure 4.23 shows the plotted results of a test run with real-life data. The hydro-thermal system comprises 25 thermal unit and 7 pumped hydro storage plants. The data of this configuration were provided by Vereinigte Energiewerke VEAG Berlin. The scenarios were constructed [5] by the following approach (cf. [GKNRW99]):

- Time Series Analysis: Fitting of a special class of non-stationary linear time series models (SARIMA) (cf. [BD96, TSP95])

[4] However, this assumption cannot be proved.

[5] This work was done by my colleagues Dr. N. Gröwe-Kuska and I. Wegner

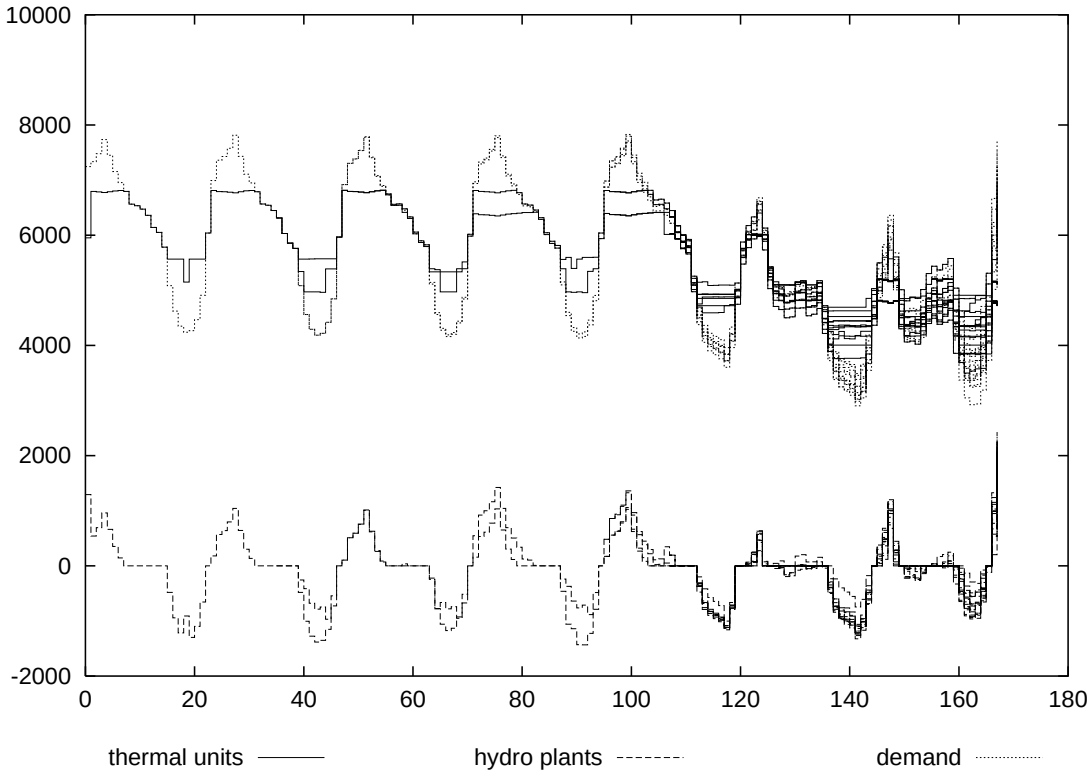

Figure 4.23: Real life example with forecast scenarios

- Construction of an initial scenario tree (cf. [Weg99]).
- Optimal reduction of the initial tree to a numerically tractable size on the base of certain Fortet-Mourier-metrics.

During the first 40 hours the forecast is almost exact. This is represented by common values for all scenarios. The decrease of information about the future realization of the demand leads to an increasing number of different scenarios.

At time period 40 the uncertainty is represented by 2 scenarios. Here, the decision process anticipates the future and leads to different schedules for the scenarios. In one scenario thermal unit are kept in operation and that energy is used to pump water uphill. The same situation can be observed at time periods 75 and 105. This stored amount of water is then used during the weekend (115−165). Here, the thermal output of the system does not vary as much as the demand. Generally, the pumped hydro storage plants are used to:

- keep the thermal units in efficient region of operation,
- prevent a cycling operation of the unit or
- to cover load peaks.

Note, the output of the compound algorithm should give just hints on necessary decisions, which have to be made right now. The schedules for a weekend, which are a result of a program run on a Monday, are neither optimal nor even feasible. Instead, the solution of problem 2.1 provides explanations, how future unknown events should be anticipated.

For example, reading Figure 4.23 leads to the insight, that a thermal unit should be kept online during time periods $42 - 47$, if the demand is above a certain level at the hour 42, because it is very likely, that the demand is higher than normal some 40 hours later. Using water, which was stored during $42 - 47$, the demand can be covered by an increased operation of the pumped hydro storage plants.

Conclusions

This thesis has treated a model for the cost optimal power scheduling of a hydro-thermal system under uncertainty of electricity demand. The model was solved by the application of stochastic Lagrangian relaxation. This approach consists in a "stochastic version" of the classical Lagrangian relaxation idea ([Lem92]) that is very popular in power optimization ([BLSP83, FK98, GMR+97, LR96, SF94, WW96, ZG88]). In difference to the application of Lagrangian relaxation to the deterministic graph-equivalent ([BLRS99]) the stochastic Lagrangian relaxation yields a different (stochastic) scalar product of the stochastic Lagrange multipliers and the left-hand side of the relaxed constraints. The numerical results justified such a modification, because the authors of [BLRS99] had to use a preconditioner in order to avoid ill-conditioning of the dual problem.

Due to the relaxation of unit coupling constraints, the mixed-integer multi-stage stochastic linear program with fixed dual parameters decomposes into a number of stochastic subproblems of lower dimension. Since only few constraints were subjected to the relaxation, the non-differentiable dual problem is of lower dimension, too.

In the presented approach the treatment of stochasticity is delayed until the solution of the subproblems. Due to the common measurability of the decision variables and the Lagrange parameters the stochastic subproblems have counterparts in a deterministic framework — as there are network flow problems and deterministic unit commitment. In this Ph.D. work a descent method for stochastic storage problems and a dynamic programming algorithm adapted to stochastic programs were developed. These solution methods can handle the stochasticity without an essential impact on the computation times. Therefore, the delayed treatment of the stochasticity is one of the major advantages of the stochastic Lagrangian relaxation.

The solver for the dual problem just provides marginal prices for the relaxed constraints. In this paper a new heuristics is developed that bases on the interplay of dual parameters and the corresponding set of preferred primal points. The heuristics performs a search along a certain path through this set. At each point an economic dispatch problem is solved by an iteration method using the last solution as a starting point. The design of the economic dispatch solver as an iteration method allows the evaluation of a sequence of schedules without solving

each problem from scratch. It was shown, that the evaluation of a sequence leads to better numerical results than a search for a feasible point. That is the price that has to be paid for the presence of pumped hydro storage plants.

The heuristics developed in this paper is able to return a stochastic solution for the optimization of a hydro-thermal power generation system under uncertainty. It was compared on deterministic problems with a compound heuristics developed by the authors of [DGM+97]. It was shown, that the new heuristics needs more computation time, but can provide tighter bounds for the duality gap.

The author of [Fel97] utilizes a different heuristics for deterministic problems. This heuristics bases on the twice-dual problem. The Lagrange parameters corresponding to the active set strategy within the bundle method are used to derive some "probabilistic" schedules ($u_{it} \in [0,1]$). Variables with value in $(0,1)$ correspond to schedules, which are neither dual preferred to be online nor preferred to be offline in the framework of this thesis. In [Fel97] a sampling method is applied in order to fix the binary variables, while in this paper a search along a path is carried out.

In the paper [BLRS99] numerical results are reported for a stochastic power scheduling problem of a hydro-thermal system. The considered system comprises 57 nuclear power plants, 15 hydro-valleys and about 60 thermal units. The solution method bases on the Lagrangian relaxation of the deterministic graph-equivalent and provides marginal prices as a result.

A stochastic model is also focused in [TBL96]. First, the problem is decomposed into single scenario problems that are solved by Lagrangian relaxation and dynamic programming. The outer progressive hedging procedure restores the nonanticipativity. When a feasible point is obtained during the iterations, then the duality gap is calculated. If the number of iterations exceeds a certain limit or the duality gap is below a threshold, the latest feasible solution is reported as the result. The same model is solved in [TKW97], but incorporating fuel constraints and spot prices. The approach is similar to that in [BLRS99], i.e. Lagrangian relaxation of the deterministic graph-equivalent. The stochastic single unit problems are similarly solved as in the present thesis. However, time dependent startup cost are not included in that paper. For updating the stochastic Lagrange parameter a subgradient method with a quadratic approximation of the dual function instead of a line-search method was used.

The numerical results in the this thesis have shown that the stochastic Lagrangian relaxation method successfully solves the stochastic power scheduling problem. In difference to the mentioned papers, the cost function of the thermal systems was piecewise linear, while the startup costs may depend on the duration the unit was down. The Lagrangian heuristics bases on the interplay of dual and primal variables and can provide a primal solution with a reasonably good objective value with a certificate of mostly less than 0.5 %.

Notations

For convenience the most used notations of this paper are given here.

Sets

Together with the sets their corresponding indices are presented.

$1, \ldots, I$ set of indices i for thermal units,

$1, \ldots, J$ set of indices j for pumped hydro storage plants,

$1, \ldots, T$ set of indices t for time periods,

Ω set of random events ω, $(\Omega, \mathcal{A}, \mathbb{P})$ is the corresponding probability space,

$\mathcal{A}$ a σ-field on Ω,

$\mathcal{F}$ a filtration, i.e. a increasing sequence of σ-fields on Ω: $\mathcal{F} = \{\mathcal{A}_t\}_{t=1}^{T}$,

$[\omega]_t \subset \Omega$ equivalence class of ω at time t,

V set of nodes $v = ([\omega]_t, t)$ of the scenario tree,

E set of arcs $e = (v, \bar{v})$ of the scenario tree,

2^A the power set: $2^A = \{B | B \subseteq A\}$,

$(\mathbb{R}, \mathcal{B})$ the space of real numbers and the family of Borel-sets.

Variables, Constants, Data, and Abbreviations

$\omega, \nu \in \Omega$ random events,

$k = ([\omega]_t, t) \in 2^{\Omega} \times \{1, \ldots, T\}$ a node in the scenario tree,

$u_i^t : \Omega \to \{0, 1\}$ binary decision for the thermal unit i and time period t, 1 means the unit is online, while 0 denotes the offline state,

$p_i^t : \Omega \to \mathbb{R}$ output level of thermal unit i and time period t,

$s_j^t : \Omega \to \mathbb{R}$ output level of hydro plant j and time period t,

$w_j^t : \Omega \to \mathbb{R}$ the amount of energy used by the pumped storage hydro plant j in the time period t for pumping water uphill,

$l_j^t : \Omega \to \mathbb{R}$ the storage level of hydro plant j in the time period t,

$d^t : \Omega \to \mathbb{R}$ the random demand of electricity,

$r^t : \Omega \to \mathbb{R}$ the requested spinning reserve,

$\lambda^t : \Omega \to \mathbb{R}^+$ the dual parameter to the demand constraint in time period t,

$\mu^t : \Omega \to \mathbb{R}^+$ the dual parameter to the reserve constraint in time period t,

π_k the probability of node k, i.e. $\pi_k = P([\omega_k]_{t_k})$,

bold variables like $\boldsymbol{d}, \boldsymbol{r}, \boldsymbol{p}, \boldsymbol{u}, \boldsymbol{s}, \boldsymbol{w}, \boldsymbol{l}, \boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}$ denote the corresponding stochastic variable without mentioning the argument ω.

Functions

$FC_i : \{0,1\} \times \mathbb{R} \to \mathbb{R}$ fuel cost function of unit i,

$SC_i^t : \{0,1\}^T \to \mathbb{R}$ start-up cost function of unit i in time period t

$D : \mathbb{R}^T \times \mathbb{R}^T \to \mathbb{R}$ the dual function,

$\hat{D}_i : \mathbb{R}^T \times \mathbb{R}^T \to \mathbb{R}$ the part of the dual function according to thermal unit i,

$\tilde{D}_j : \mathbb{R}^T \times \mathbb{R}^T \to \mathbb{R}$ the part of dual function according to storage plant j,

$Succ(k) : V \to 2^V$ the set of all immediate successors of the node k.

Terms

scenario A scenario is just an element ω of the abstract probability space Ω.

g-subtree A generalized subtree (short g-subtree) is a graph (V', E'), which is a subgraph of a certain graph (in this paper the scenario tree) , i.e.: $(V', E') \subseteq (V, E)$, and which is a tree for itself. *The difference to the well-known subtree is, that the leaves of a g-subtree do not have to be leaves in the originating tree.*

List of Figures

1.1 Example of a scenario tree 16
1.2 Scenario tree and deterministic LP-equivalent 19
1.3 Flow chart for the Nested Benders Decomposition Algorithm . . . 28

2.1 Example of a load process 32
2.2 Example: one stage, one unit 45

3.1 Example of a simple direction 57
3.2 Deterministic Dynamic Programming 71
3.3 Stochastic Dynamic Programming 72

4.3 Computation times of NOA 3.0 on a HP-J280 76
4.4 Number of steps of EXCHA 78
4.5 Computation time of EXCHA 79
4.6 Computation times of EXCHA 80
4.7 Feasible transactions 81
4.8 The complete dynamic programming graph for a unit with 4 hours must-on time, 3 hours must-off time, and 5 hours cooling time. . . 81
4.9 The exponential startup curve 82
4.10 The reduced dynamic programming graph 83
4.15 Cycle Unit in a high load week starting on Friday morning 88
4.16 Cycle Unit in a low load week starting on Wednesdays noon . . . 89
4.17 Objective value and feasibility of the chosen sequence 90
4.18 Comparison of computing times 93
4.19 Comparison of upper bounds for the duality gaps 94
4.20 Number of ECDISP iterations against the number of scenarios . . 99
4.21 Computation times against the number of scenarios for the ECDISP algorithm 100
4.22 Flow Chart for the Complete Algorithm 103
4.23 Real life example with forecast scenarios 105

List of Tables

2.1 Dimension of the mixed-integer LP depending on the number of scenarios with T=168, I=25 and J=7 37
2.2 Dimension of the dual problem in comparison to the primal problem 43

4.1 Computation time in seconds of CPLEX for the small example . . 97
4.2 Computation time in seconds of CPLEX for the bigger example . 98
4.3 Ratio of the computation times of CPLEX to ECDISP 98
4.4 Results for the compound algorithm 102

Bibliography

[BD96] P.J. Brockwell and R.A. Davis. *Introduction to Time Series and Forecasting*. Springer, New York, 1996.

[Bel57] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, N.J., 1957.

[Ben62] J.F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[Ber82] D.P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, 1982.

[Ber87] D.P. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models*. Prentice Hall, Inc., Eglewood Cliffs, N.J., 1987.

[BGK+82] B. Bank, J. Guddat, D. Klatte, B. Kummer, and K. Tammer. *Non-Linear Parametric Optimization*. Akademie-Verlag, 1982.

[Bir88] J.R. Birge. An L-shaped method computer code for multistage stochastic linear programs. In *Numerical techniques for stochastic optimization, Springer Ser. Comput. Math. 10, 255-266*, 1988.

[Bir97] J.R. Birge. Stochastic programming computation and applications. *INFORMS Journal on Computing*, 9(2):111–133, 1997.

[BL88] J.R. Birge and F.V. Louveaux. A multicut algorithm for two-stage stochastic linear programs. *European J. Oper. Res.*, 34(3):384–392, 1988.

[BL97] J.R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Series in Operations Research. Springer-Verlag, New York, 1997.

[BLRS99] L. Bacaud, C. Lemaréchal, A. Renaud, and C. Sagstizábal. Disaggregated bundle methods in stochastic optimal power management. *Computational Optimization and Applications*, 1999. submitted.

[BLSP83] D.P. Bertsekas, G.S. Lauer, N.R. Sandell, and T.A. Posbergh. Optimal short-term scheduling of large-scale power systems. *IEEE Transactions on AC*, 28(1):1–11, January 1983.

[BTL98] J.R. Birge, S. Takriti, and E. Long. Intelligent unified control of unit commitment and generation allocation. Final report, Department of Industrial and Operations Engineering, The University of Michigan, 1998.

[Car98] C.C. Carøe. *Decomposition in Stochastic Integer Programming.* PhD thesis, Department of Operations Research, University of Copenhagen, Denmark, 1998.

[CNRS99] C.C. Carøe, M.P. Nowak, W. Römisch, and R. Schultz. Power scheduling in a hydro-thermal system under uncertainty. *Proceedings of the 13th Power Systems Computation Conference (Trondheim, Norway)*, 2:1086–1092, 1999.

[CS98] C.C. Carøe and R. Schultz. A two-stage stochastic program for unit commitment under uncertainty in a hydro-thermal power system. *Preprint 98-13, DFG-Schwerpunktprogramm "Echtzeit-Optimierung großer Systeme"*, 1998.

[CS99] C.C. Carøe and R. Schultz. Dual decomposition in stochastic integer programming. *Operation Research Letters*, 24:37–45, 1999.

[DCW] J. Dupačová, G. Consigli, and S.W. Wallace. Scenarios for multistage stochastic programs. *Annals of Operations Research.* submitted.

[DGM+97] D. Dentcheva, R. Gollmer, A. Möller, W. Römisch, and R. Schultz. Solving the unit commitment problem in power generation by primal and dual methods. In *Progress in Industrial Mathematics at ECMI 96*, pages 332 – 339, Stuttgart, 1997. Teubner.

[Dup92] J. Dupačová. Stochastic programming models in banking & portfolio optimization under uncertainty. In W. Runggaldier, editor, *Stochastic Processes: Applications in Mathematical Economics - Finance*, Applied Mathematics Monographs 5, pages 19–20. C.N.R., 1992.

[EG92] Y.M. Ermoliev and A.A. Gaivoronski. Stochastic quasigradient methods for optimization of discrete event systems. *Ann. Oper. Res.*, 39:1–39, 1992.

[EK95] L. F. Escudero and P. V. Kamesam. On solving stochastic production planning problems via scenario modelling. *Top*, 3(1):69–95, 1995.

[EKKW93] Laureano F. Escudero, Pasumarti V. Kamesam, Alan J. King, and Roger J.-B. Wets. Production planning via scenario modelling. *Ann. Oper. Res.*, 43(1-4):311–335, 1993. Applied mathematical programming and modelling (Uxbridge, 1991).

[Erm88] Y.M. Ermoliev. Stochastic quasigradient methods. In *Numerical techniques for stochastic optimization*, volume 10 of *Springer Ser. Comput. Math.*, pages 141–185. Springer, Berlin, 1988.

[EW88] Y.M. Ermoliev and R.J-B. Wets. *Numerical Techniques for Stochastic Optimization*. Springer-Verlag, 1988.

[Fel97] S. Feltenmark. *On Optimization of Power Production*. PhD thesis, Department of Mathematics, Royal Institute of Technology, Stockholm, Sweden, 1997.

[FK98] S. Feltenmark and K.C. Kiwiel. Dual applications of proximal bundle methods, including Lagrangian relaxation of nonconvex problems. Report TRITA/MAT-98-OS1, Department of Mathematics, Royal Institute of Technology, Stockholm, January 1998. revised November 1998 and to appear in *SIAM Journal on Optimization*.

[Gas90] H. I. Gassmann. Mslip: A computer code for the multistage stochastic linear programming problem. *Mathematical Programming*, 47:407–423, 1990.

[GKNRW99] N. Gröwe-Kuska, M.P. Nowak, W. Römisch, and I. Wegner. Optimierung eines hydro-thermischen Kraftwerkssystems unter Ungewissheit. *Optimierung in der Energieversorgung. Planungsaufgaben in liberalisierten Energiemärkten, VDI-Berichte*, 1508:147–157, 1999.

[GMN+99] R. Gollmer, A. Möller, M.P. Nowak, W. Römisch, and R. Schultz. Primal and dual methods for unit commitment in a hydro-thermal power system. *Proceedings of the 13th Power Systems Computation Conference (Trondheim, Norway)*, 2:724–730, 1999.

[GMR+97] R. Gollmer, A. Möller, W. Römisch, R. Schultz, G. Schwarzbach, and J. Thomas. Optimale Blockauswahl bei der Kraftwerkseinsatzplanung der VEAG. *Optimierung in der Energieversorgung II*, VDI-Berichte 1352, Düsseldorf:71–85, 1997.

[GNRS99] R. Gollmer, M.P. Nowak, W. Römisch, and R. Schultz. Unit commitment in power generation - a basic model and some extensions. *Schriftenreihe des Fachbereichs Mathematik, SM-DU-445, Gerhard-Mercator-Universität Duisburg*, April 1999. (to appear in Annals of Operations Research).

[GRS92] J. Guddat, W. Römisch, and R. Schultz. Some applications of mathematical programming techniques in optimal power dispatch. *Computing*, 49:193–200, 1992.

[HS91] J.L. Higle and S. Sen. Stochastic decomposition: An algorithm for two-stage linear programs with recourse. *Mathematics of Operations Research*, 16:650 – 669, 1991.

[HS96] J.L. Higle and S. Sen. Stochastic decomposition: A statistical method for large scale stochastic linear programming. *Kluwer Academic Publishers, Dordrecht*, 1996.

[HUL96a] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms I*, volume 305 of *A Series of Comprehensive Studies in Mathematics*. Springer-Verlag, 1996.

[HUL96b] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms II*, volume 306 of *A Series of Comprehensive Studies in Mathematics*. Springer-Verlag, 1996.

[Kiw90] K.C. Kiwiel. Proximity control in bundle methods for convex nondifferentiable minimization. *Mathematical Programming*, 46:105–122, 1990.

[Kiw95] K.C. Kiwiel. Approximations in proximal bundle methods and decomposition of convex programs. *Journal of Optimization Theory and Applications*, 84:529–548, 1995.

[Kiw94] K.C. Kiwiel. *User's Guide for NOA 2.0/3.0: A Fortran package for convex nondifferentiable optimization*. Polish Academy of Sciences, Systems Research Institute, Warsaw (Poland), 1993/94.

[KW94] P. Kall and S.W. Wallace. *Stochastic Programming*. John Wiley and Sons, 1994.

[Lem92] C. Lemaréchal. Lagrangian decomposition and nonsmooth optimization: bundle algorithm, prox iteration, augmented Lagrangian. In F. Gianessi, editor, *Nonsmooth Optimization Methods and Applications*, pages 201–216. Gordon and Breach, 1992.

[LP66] E.S. Levitin and B.T. Polyak. Constraint minimization methods. *Computational Mathematics and Mathematical Physics*, 6(5):1–50, 1966.

[LR96] C. Lemaréchal and A. Renaud. Dual equivalent convex and non-convex problems. Research report, INRIA, Rocquencourt, 1996.

[LTZ98] P.B. Luh, R.N. Tomastik, and D. Zhang. An algorithm for solving the dual problem of hydrothermal scheduling. *IEEE Transactions on Power Systems*, 13:593–600, 1998.

[LW96] A. Løkketangen and D.L. Woodruff. Progressive hedging and tabu search applied to mixed integer (0,1) multi-stage stochastic programming. *Journal of Heuristics*, 2:111–128, 1996.

[Nie98] S. Nießen. *Kraftwerkseinsatz- und Handelsplanung im liberalisierten Strommarkt.* PhD thesis, Institut für Elektrische Anlagen und Energiewirtschaft, Forschungsgesellschaft Energie an der RWTH Aachen, 1998.

[Now96] M.P. Nowak. A fast descent method for the hydro storage subproblem in power generation. Working Paper 96-109, International Institute for Applied Systems Analysis, October 1996.

[NPR98] V.I. Norkin, G. Pflug, and A. Ruszczyński. A branch and bound method for stochastic global optimization. *Mathematical Programming*, 83:425–450, 1998.

[NR98] M.P. Nowak and W. Römisch. Stochastic Lagrangian relaxation applied to power scheduling in a hydro-thermal system under uncertainty. *Preprint 98-36, DFG-Schwerpunktprogramm "Echtzeit-Optimierung großer Systeme"*, 1998. submitted to *Annals of Operations Research.*

[PCW00] A.B. Philpott, M. Craddock, and H. Waterer. Hydro-electric unit commitment subject to uncertain demand. *European Journal of OR*, 125:410 – 425, 2000.

[Pol97] E. Polak. *Optimization: Algorithms and Consistent Approximations*, volume 124 of *Applied Mathematical Sciences.* Springer, 1997.

[PP91] M.V.F Pereira and L.M.V.G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52:359–375, 1991.

[RC99] N. Jiménez Redondo and A.J. Conejo. Short-term hydro-thermal coordination by Lagrangian relaxation: Solution of the dual problem. *IEEE Transactions on Power Systems*, 14(1):89–95, February 1999.

[Rus97] A. Ruszczyński. Decomposition methods in stochastic programming. *Mathematical Programming (Series B)*, 79(1-3):333–353, 1997. Lectures on mathematical programming (ismp97) (Lausanne, 1997).

[RW76] R.T. Rockafellar and R.J.-B. Wets. Nonanticipativity and L^1-martingales in stochastic optimization problems. *Math. Prog. Study*, 6:170–187, 1976.

[RW78] R.T. Rockafellar and R.J.-B. Wets. The optimal recourse problem in discrete time: L^1-multipliers for inequality contraints. *SIAM Journal on Control and Optimization*, (16):16–36, 1978.

[RW91] R.T. Rockafellar and R.J.-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147, 1991.

[SF94] G.B. Sheble and G.N. Fahd. Unit commitment literature synopsis. *IEEE Transactions on Power Systems*, 9:128–135, 1994.

[TBL96] S. Takriti, J.R. Birge, and E. Long. A stochastic model for the unit commitment problem. *IEEE Transactions on Power Systems*, 11:1497–1508, 1996.

[TKW97] S. Takriti, B. Krasenbrink, and L. S.-Y. Wu. Incorporating fuel contraints and electricity spot prices into the stochastic unit commitment problem. IBM Research Report RC 21066(12/29/1997), IBM Research Division, December 1997.

[TSP95] Wolfram Research. *Time Series Pack: Reference and Users Guide*, 1995.

[VW69] R. Van Slyke and R.J.-B. Wets. L-shaped linear programs with application to optimal control and stochastic optimization. *SIAM Journal on Applied Mathematics*, 17:638–663, 1969.

[Weg99] I. Wegner. Erzeugung von Szenariobäumen für die Kraftwerks-Einsatzoptimierung. Master's thesis, Humboldt-University to Berlin, Humboldt-University to Berlin, Department for Applied Mathematics, Berlin 10099, Germany, 1999.

[Wet89] R.J.-B. Wets. Stochastic programming. In *Optimization*, volume 1 of *Handbooks in Operations Research and Management Science*, pages 573–629. North Holland, Amsterdam, 1989.

[WW96] A.J. Wood and B.F. Wollenberg. *Power Generation, Operation, and Control*. Wiley, New York, second edition, 1996.

[ZG88] F. Zhuang and F.D. Galiana. Towards a more rigorous and practical unit commitment by Lagrangian relaxation. *IEEE Transactions on Power Systems*, 3:763 – 773, 1988.

Index

$\mathcal{F} = \mathcal{F}(\boldsymbol{x})$, 13

relatively complete recourse problem, 24

Bundle Method, 74

coloring, 85
complete recourse problem, 24

deterministic
  graph-equivalent, 20
  LP-equivalent, 18
dual information, 46

equivalence class, 15

facet search, 50
feasibility cut, 26
filtration, 13
  generated filtration, 13
flexibility, 34
  rolling time horizon, 35

g-subtree, 110

information constraint, 15
information structure, 12

Kiwiel, 74

Lagrangian reduction, 46, 50
Lagrangian relaxation, 41

Markovian structure, 23
master problem, 21
method
  decomposition
    dual, 21
    nested Benders, 21, 22
    primal, 21
    scenario decomposition, 22
  L-shaped, 21
  stochastic branch-and-bound, 21
  stochastic quasigradient, 21
multi-stage stochastic linear program, 17
multi-stage stochastic problem, 12

necessary optimality condition, 49
network flow problem, 41
nonanticipative process, 14
Nonanticipativity, 14
Nonanticipativity-constraints, 34

optimality cut, 27
optimality of crossover points, 46

progressive hedging, 22
proximal term, 22

random event, 10
random variable, 11
realization, 10
recourse function, 23
recourse matrix, 23
recourse model, 11
resource state variable, 57

scenario, 10
scenario decomposition, 22
scenario tree
  definition, 16
scenario tree , 15–17
SDDP, 27
simple recourse, 24

solution
    admissible, 10
    feasible, 10
stochastic Lagrangian relaxation, 36
stochastic process with discrete time, 12
strong duality, 43

technology matrix, 23
two-stage model, 11

unit commitment problem, 41
upper semi-continuous according to Berge, 49

VEAG-Problem, 36

Lebenslauf

Name:	Matthias Peter Nowak
09/1984 - 08/1986	Abitur, Spezialschule für Mathematik/Physik der Humboldt-Universität zu Berlin
09/1989 - 04/1996	Studium an der Humboldt-Universität zu Berlin in der Fachrichtung Mathematik
05/1996 - 07/1999	Forschungsstudent im Graduiertenkolleg "Geometrie und Nichtlineare Analysis"
08/1999 - 12/1999	Wissenschaftlicher Mitarbeiter an der Universität Duisburg Fachbereich 11 / Mathematik Forschungsgruppe Prof. R. Schultz

Selbständigkeitserklärung

Hiermit erkläre ich, die vorliegende Arbeit selbständig ohne fremde Hilfe verfaßt zu haben und nur die angegebene Literatur und Hilfsmittel verwendet zu haben.

Matthias Peter Nowak
23. Dezember 1999